

Applicants: Perry A. Caro et al.

Serial No.:

Filed : July 23, 1999

Title : COMPUTER GENERATION OF DOCUMENTS USING LAYOUT ELEMENTS AND CONTENT ELEMENTS

APPENDIX A

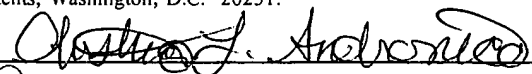
227 Pages



"EXPRESS MAIL" Mailing Label Number EL22467321405

Date of Deposit 7.23.99

I hereby certify under 37 CFR 1.10 that this correspondence is being deposited with the United States Postal Service as "Express Mail Post Office To Addressee" with sufficient postage on the date indicated above and is addressed to the Assistant Commissioner for Patents, Washington, D.C. 20231.


Christina L. Andronico

package com.adobe.cbl

Interface Index

- AudioDynCModel
- CModel
- ColorStylesLModel
- ContentPortfolio
- DirectBinding
- DocGfxCModel
- DynCModel
- DynStylesLModel
- DynamicsLModel
- FlowTxtCModel
- FontStylesLModel
- GfxCModel
- GfxStylesLModel
- LModel
- LayeredGfxCModel
- LayoutPortfolio
- LinkCModel
- Measure
- Model
- MultiGfxCModel
- PFElement
- PlacedTxtCModel
- PlacementLModel
- Portfolio
- RasterGfxCModel
- Referer
- SiteEnumeration
- StylesLModel
- TriDGfxCModel

- TxtCModel
- TxtStylesLModel
- VectorGfxCModel
- VideoDynCModel

Class Index

- APIHelper
- BindingSpec
- CompBlock
- CompSequence
- Corner
- DimensionPro
- DocSpec
- ElementBinding
- Env
- IntervalPro
- ModelBinding
- ModelHelper
- PFElementWrapper
- PFEnumeration
- PlacementBinding
- PortfolioType
- RectPro
- Site
- SiteUsage
- StyleBinding
- Template
- TemplateUsage
- VSiteEnumeration

Exception Index

- CBLEException
- CommitFailed_CBLEException
- LimitCheckException

- OpenFailed_CBLEException
- UnsupportedFeature_CBLEException

Package Index

- package com.adobe.cbl

Class Hierarchy

- classjava.lang.Object
 - classcom.adobe.cbl. APIHelper
 - interfacecom.adobe.cbl. AudioDynCModel(extendscom.adobe.cbl. DynCModel)
 - classcom.adobe.cbl. BindingSpec(implementscom.adobe.cbl. Referer,java.lang.Cloneable)
 - interfacecom.adobe.cbl. CModel(extendscom.adobe.cbl. Model)
 - interfacecom.adobe.cbl. ColorStylesLModel(extendscom.adobe.cbl. StylesLModel)
 - classcom.adobe.cbl. CompBlock
 - classcom.adobe.cbl. CompSequence(implementsjava.lang.Cloneable)
 - interfacecom.adobe.cbl. ContentPortfolio(extendscom.adobe.cbl. Portfolio)
 - classcom.adobe.cbl. Corner
 - classjava.util.Dictionary
 - classjava.util.Hashtable(implementsjava.lang.Cloneable,java.io.Serializable)
 - classcom.adobe.cbl. Env
 - classcom.adobe.cbl. DimensionPro
 - interfacecom.adobe.cbl. DirectBinding
 - interfacecom.adobe.cbl. DocGfxCModel(extendscom.adobe.cbl. GfxCModel)
 - classcom.adobe.cbl. DocSpec
 - interfacecom.adobe.cbl. DynCModel(extendscom.adobe.cbl. CModel)
 - interfacecom.adobe.cbl. DynStylesLModel(extendscom.adobe.cbl. StylesLModel)
 - interfacecom.adobe.cbl. DynamicsLModel(extendscom.adobe.cbl. LModel)
 - classcom.adobe.cbl. ElementBinding
 - interfacecom.adobe.cbl. FlowTxtCModel(extendscom.adobe.cbl. TxtCModel)
 - interfacecom.adobe.cbl. FontStylesLModel(extendscom.adobe.cbl. StylesLModel)
 - interfacecom.adobe.cbl. GfxCModel(extendscom.adobe.cbl. CModel)
 - interfacecom.adobe.cbl. GfxStylesLModel(extendscom.adobe.cbl. StylesLModel)
 - classcom.adobe.cbl. IntervalPro
 - interfacecom.adobe.cbl. LModel(extendscom.adobe.cbl. Model)
 - interfacecom.adobe.cbl. LayeredGfxCModel(extendscom.adobe.cbl. GfxCModel)
 - interfacecom.adobe.cbl. LayoutPortfolio(extendscom.adobe.cbl. Portfolio)

- interface com.adobe.cbl.LinkCModel (extends com.adobe.cbl.CModel)
- interface com.adobe.cbl. Measure
- interface com.adobe.cbl. Model
- class com.adobe.cbl. ModelBinding
- class com.adobe.cbl. ModelHelper
- interface com.adobe.cbl. MultiGfxCModel (extends com.adobe.cbl. GfxCModel)
- interface com.adobe.cbl. PFElement
- class com.adobe.cbl. PFElementWrapper (implements com.adobe.cbl. PFElement)
- class com.adobe.cbl. PFEnumeration (implements java.util.Enumeration)
- interface com.adobe.cbl. PlacedTxtCModel (extends com.adobe.cbl. TxtCModel)
- class com.adobe.cbl. PlacementBinding (implements com.adobe.cbl. DirectBinding)
- interface com.adobe.cbl. PlacementLModel (extends com.adobe.cbl. LModel)
- interface com.adobe.cbl. Portfolio
- class com.adobe.cbl. PortfolioType
- interface com.adobe.cbl. RasterGfxCModel (extends com.adobe.cbl. GfxCModel)
- class com.adobe.cbl. RectPro
- interface com.adobe.cbl. Referer
- class com.adobe.cbl. Site
- interface com.adobe.cbl. SiteEnumeration (extends java.util.Enumeration)
- class com.adobe.cbl. SiteUsage
- class com.adobe.cbl. StyleBinding (implements com.adobe.cbl. DirectBinding)
- interface com.adobe.cbl. StylesLModel (extends com.adobe.cbl. LModel)
- class com.adobe.cbl. Template
- class com.adobe.cbl. TemplateUsage
- class java.lang.Throwable (implements java.io.Serializable)
- class java.lang.Exception
 - class com.adobe.cbl. CBLEException
 - class com.adobe.cbl. CommitFailed_CBLEException
 - class com.adobe.cbl. OpenFailed_CBLEException
 - class com.adobe.cbl. UnsupportedFeature_CBLEException
 - class java.lang.RuntimeException
 - class com.adobe.cbl. LimitCheckException
- interface com.adobe.cbl. TriDGfxCModel (extends com.adobe.cbl. GfxCModel)
- interface com.adobe.cbl. TxtCModel (extends com.adobe.cbl. CModel)

- interface com.adobe.cbl.TxtStylesLModel (extends com.adobe.cbl.StylesLModel)
- class com.adobe.cbl.VSiteEnumeration(implements com.adobe.cbl.SiteEnumeration)
- interface com.adobe.cbl.VectorGfxCModel(extends com.adobe.cbl.GfxCModel)
- interface com.adobe.cbl.VideoDynCModel(extends com.adobe.cbl.DynCModel)

Index of all Fields and Methods

A

addAfter(Vector, Object, Object).Staticmethodinclasscom.adobe.cbl. BindingSpec
Vectorinsertconveniencemethod.

addCompBlock(CompBlock, CompBlock).Methodinclasscom.adobe.cbl. CompSequence
Addacompositionblock.

addCompSeq(CompSequence, CompSequence).Methodinclasscom.adobe.cbl. BindingSpec
Addadirectbindingcompositionsequenceinorder.

addDirectBinding(DirectBinding, DirectBinding).Methodinclasscom.adobe.cbl. CompBlock
Adddirectbinding.

addElementBinding(ElementBinding).Methodinclasscom.adobe.cbl. BindingSpec
Addanelementbinding.

Orderisnotmaintained.

addModelBinding(ModelBinding).Methodinclasscom.adobe.cbl. BindingSpec
Addamodelbinding.

Orderisnotmaintained.

addOrderedStyles(CompBlock, CompBlock).Methodinclasscom.adobe.cbl. CompBlock
Addorderedstyles.

APIHelper(Writer).Constructorforclasscom.adobe.cbl. APIHelper
ConstructwithaWriter.

ARTICLE.Staticvariableininterfacecom.adobe.cbl. FlowTxtCModel
Asequenceofsections,withanoptionalheading(title).

ARTTEXT. Static variable in interface com.adobe.cbl.PlacedTxtCModel
Identifiestextart,suchastextonacurve.

.Staticvariableininterfacecom.adobe.cbl. MultiGfxCModel
Identifiesanartworkview.

B

BindingSpec(). Constructor for class com.adobe.cbl. BindingSpec
CreateanemptyBindingSpec.

BindingSpec(String). Constructor for class com.adobe.cbl. BindingSpec
CreateanemptyBindingSpecwithRID.

BOTTOM_LEFT. Static variable in class com.adobe.cbl. Corner
xincreasestotheright,yincreasesup.

BOTTOM_RIGHT. Static variable in class com.adobe.cbl. Corner
xincreasestothelleft,yincreasesup.

BREAK. Static variable in interface com.adobe.cbl. PlacementLModel
Identifiesa(line)break.

C

(Site,Site).Methodininterfacecom.adobe.cbl. Portfolio
Addasitetoagroupsite.

Groupsarenotallovedtocontainself-referencesatanylevel.

CBL_checkSite(Site). Method in interface com.adobe.cbl. Portfolio
CheckifSiteisvalid,returnitsusage.

CBL_checkSitePFType(Site). Method in interface com.adobe.cbl. DirectBinding
CheckbindingforSiteandPortfoliotype.

UsedtodiscoverifaparticularSiteisusedinthisbinding,andwhattypeofPortfolioitis
expected to be in.

CBL_checkSitePFTType(Site).Methodin class com.adobe.cbl. PlacementBinding
Check binding for Site and Portfolio type.

Used to discover if a particular Site is used in this binding, and what type of Portfolio it is expected to be in.

CBL_checkSitePFTType(Site).Methodin class com.adobe.cbl. StyleBinding
Check binding for Site and Portfolio type.

Used to discover if a particular Site is used in this binding, and what type of Portfolio it is expected to be in.

CBL_close().Methodin interface com.adobe.cbl. Portfolio
Close the portfolio.

Can be used for underlying file management, DB session unlocking, etc.

CBL_commit().Methodin interface com.adobe.cbl. Portfolio
Commit the portfolio.

Can be used to commit a transaction, do a Save of the file, etc.

CBL_deleteElementAttr(String).Methodin interface com.adobe.cbl. PFElement
Delete an attribute from this element.

CBL_deleteElementAttr(String).Methodin class com.adobe.cbl. PFElementWrapper

CBL_deletePortfolioAttr(String).Methodin interface com.adobe.cbl. Portfolio
Delete an attribute from this portfolio.

CBL_elementAttributes().Methodin interface com.adobe.cbl. PFElement
Enumerate all of the attribute names on this element.

CBL_elementAttributes().Methodin class com.adobe.cbl. PFElementWrapper

(Site).Methodin interface com.adobe.cbl. PFElement
List all the Sites associated with this element.

(Site).Methodin class com.adobe.cbl. PFElementWrapper

(String).Methodin interface com.adobe.cbl. Portfolio
List elements by an informal description.



CBL_enumGroups().Methodin interface com.adobe.cbl. Portfolio

Get all groups sites.

Groups may have multiple binding sites, but this method only returns one site per group.

CBL_enumModelRestrictions().Methodin interface com.adobe.cbl. Portfolio

Get all model bindings imposed on this Portfolio instance.

CBL_enumRIDs().Methodin interface com.adobe.cbl. Portfolio

List all known Referer identifiers.

CBL_enumSites().Methodin interface com.adobe.cbl. Portfolio

List all the binding sites in this portfolio.

CBL_enumSitesInGroup(Site).Methodin interface com.adobe.cbl. Portfolio

Get all sites in a group.

(PFEElement).Methodin interface com.adobe.cbl. PFEElement

List all the data (sub-content) sites associated with this element.

(PFEElementWrapper).Methodin class com.adobe.cbl. PFEElementWrapper

CBL_estimateDataSize(Site).Methodin interface com.adobe.cbl. PFEElement

Estimate the size of the data.

The size returned is a conservative count.

CBL_estimateDataSize(Site).Methodin class com.adobe.cbl. PFEElementWrapper

CBL_getBytes(String, Site).Methodin interface com.adobe.cbl. PFEElement

Get element data as an array of bytes.

CBL_getBytes(String, Site).Methodin class com.adobe.cbl. PFEElementWrapper

CBL_getChild(PFEElement).Methodin interface com.adobe.cbl. PFEElement

Get a child in sequence.

Get the child after the specified child.

CBL_getChild(PFEElement).Methodin class com.adobe.cbl. PFEElementWrapper

CBL_getContainerElement(Site).Methodin interface com.adobe.cbl. Portfolio

Get the element that contains sub-content associated with Site.

CBL_getDataFormat(Site).Methodininterfacecom.adobe.cbl. PFElement
Getthepreferredformatforunstructureddata.

CBL_getDataFormat(Site).Methodinclasscom.adobe.cbl. PFElementWrapper

CBL_getDstSites().Methodininterfacecom.adobe.cbl. DirectBinding
Getdestinationsites.

Theinterpretationofthesedestinationsitesdependsontheclassimplementingthisinterface:e.g.,
forPlacementBindingsthesitesarethelayoutplacementdirectivesforthecorrespondingcontents

CBL_getDstSites().Methodinclasscom.adobe.cbl. PlacementBinding
Getplacementsitesfromalayoutportfolio.

CBL_getDstSites().Methodinclasscom.adobe.cbl. StyleBinding
Getdestinationsites.

Thesearethesitesthatarethetargetsofthestyletobeapplied.

CBL_getElementAttr(String).Methodininterfacecom.adobe.cbl. PFElement
Getarbitraryattributeontheelement.

CBL_getElementAttr(String).Methodinclasscom.adobe.cbl. PFElementWrapper

CBL_getElementBySite(Site,String).Methodininterfacecom.adobe.cbl. Portfolio
GettheelementassociatedwithaSite.

CBL_getInputStream(String,Site).Methodininterfacecom.adobe.cbl. PFElement
GetelementdataasanInputStream(bytes).

Validforallmodels.

CBL_getInputStream(String,Site).Methodinclasscom.adobe.cbl. PFElementWrapper

CBL_getInts(String,Site).Methodininterfacecom.adobe.cbl. PFElement
Getelementdataasanarrayofints(32-bit).

Validforallmodels.

CBL_getInts(String,Site).Methodinclasscom.adobe.cbl. PFElementWrapper

CBL_getNickname(). Method in interface com.adobe.cbl.PFElement

Getinformaldescription.

IMPLEMENTATIONNOTE: Ifyourimplementationsupportsthegeneralattributemethods listedabove,youareencouragedtoimplementinformaldescriptionswithanattributenamed "description.informal"(youcanusethe **NICKNAME**constantdefinedabove.)

CBL_getNickname().Methodinclasscom.adobe.cbl. PFElementWrapper

CBL_getObject(Class,Site).Methodininterfacecom.adobe.cbl. PFElement

GetelementdataasanObject.

ClassaClassdependsontheelement'smodel:

GfxCModel

RectPro,returnsthegraphicssub-rectangle.

StyleLModel

Anapplicationspecificstyleobject.

CBL_getObject(Class,Site).Methodinclasscom.adobe.cbl. PFElementWrapper

CBL_getObjectType(Site).Methodininterfacecom.adobe.cbl. PFElement

Getthepreferredobjecttypeofstructureddata.

CBL_getObjectType(Site).Methodinclasscom.adobe.cbl. PFElementWrapper

CBL_getOrigSite().Methodininterfacecom.adobe.cbl. PFElement

GettheSiteusedtoretrievethiselement.

CBL_getOrigSite().Methodinclasscom.adobe.cbl. PFElementWrapper

GettheSiteusedtoretrievethiselement.

Thismethodusesinstancedataanddoesnotreferencetheoriginalelement.

CBL_getParent().Methodininterfacecom.adobe.cbl. PFElement

Getparent.

CBL_getParent().Methodinclasscom.adobe.cbl. PFElementWrapper

CBL_getPortfolio().Methodininterfacecom.adobe.cbl. PFElement

GetthePortfoliothatthiselementcamefrom.

CBL_getPortfolio(). Method in class com.adobe.cbl.PFElementWrapper

(String).Methodininterfacecom.adobe.cbl. Portfolio
Getanattributeonthisportfolio.

CBL_getPrefCModel().Methodininterfacecom.adobe.cbl. ContentPortfolio

Getthepreferredcontentmodel.

CBL_getPrefLModel().Methodininterfacecom.adobe.cbl. LayoutPortfolio

Getthepreferredlayoutmodel

(String).Methodininterfacecom.adobe.cbl. Portfolio
Getaproperty.

CBL_getReader(String,Site).Methodininterfacecom.adobe.cbl. PFElement

GetelementdataasaReader(Characters).

ValidforTxtCModelelementsonly.

IMPLEMENTATIONNOTE: AllPFElementimplementationsarerequiredtoimplementthis
methodforTxtCModels.

CBL_getReader(String,Site).Methodinclasscom.adobe.cbl. PFElementWrapper

()).Methodininterfacecom.adobe.cbl. Portfolio
GetRefererforthisinstance.

Refererisaninterfaceimplementedbysomeobjectthatholdsreferencestothisportfolio.

CBL_getRequired().Methodininterfacecom.adobe.cbl. DirectBinding

Checkifbindingisrequired.

CBL_getRequired().Methodinclasscom.adobe.cbl. PlacementBinding

Checkifbindingisrequired.

CBL_getRequired().Methodinclasscom.adobe.cbl. StyleBinding

Checkifbindingisrequired.

CBL_getRoot().Methodininterfacecom.adobe.cbl. Portfolio

GetrootPORTFOLIOelement.

AllportfoliosmaycontainreferencestootherportfolioswiththeModel.PORTFOLIOelement,

which may be nothing more than a holder for top-level elements.

CBL_getSemanticName(int).Method in interface com.adobe.cbl. PFElement
Get the semantic name used to retrieve this element.

CBL_getSemanticName(int).Method in class com.adobe.cbl. PFElementWrapper
Get the semantic name used to retrieve this element.

This method uses instance data and does not reference the original element.

CBL_getSrcSites().Method in interface com.adobe.cbl. DirectBinding
Get source sites.

The interpretation of these source sites depends on the class implementing this interface: e.g., for PlacementBinding the sites are the content to be placed in the layout somewhere.

CBL_getSrcSites().Method in class com.adobe.cbl. PlacementBinding
Get content sites from a content portfolio.

These are the content to be placed according to the layout sites defined in this binding.

CBL_getSrcSites().Method in class com.adobe.cbl. StyleBinding
Get style sites from a layout portfolio.

CBL_getString(String, Site).Method in interface com.adobe.cbl. PFElement
Get element data as a String.

Valid for TxtCModel and LinkCModel elements only.

CBL_getString(String, Site).Method in class com.adobe.cbl. PFElementWrapper

CBL_getSubRange(Site).Method in interface com.adobe.cbl. PFElement
Get range used to define sub-content.

Valid for TxtCModel elements only.

CBL_getSubRange(Site).Method in class com.adobe.cbl. PFElementWrapper

CBL_getSubStyleNames().Method in interface com.adobe.cbl. PFElement
List the supported sub-style names.

CBL_getSubStyleNames().Method in class com.adobe.cbl. PFElementWrapper

CBL_hasChildren(). Method in interface com.adobe.cbl.PFElement
Anychildren?

CBL_hasChildren(). Method in class com.adobe.cbl. PFElementWrapper

CBL_hasData(). Method in interface com.adobe.cbl. PFElement
Check if data can be retrieved from this element.

CBL_hasData(). Method in class com.adobe.cbl. PFElementWrapper

CBL_imposeSite(Site). Method in interface com.adobe.cbl. PFElement
Impose a site on this element.

Use this method when you want to impose an existing site (e.g., from a BindingSpec) on an element.

CBL_imposeSite(Site). Method in class com.adobe.cbl. PFElementWrapper

CBL_imposeSiteOnGroup(Site, Site). Method in interface com.adobe.cbl. Portfolio
Impose a site on an existing group.

Use this method when you want to impose an existing site (e.g., from a BindingSpec) on a group.

CBL_isAsIntended(). Method in interface com.adobe.cbl. PFElement
Test this element to see if it has the intended semantic name.

Elements are normally acquired by doing a lookup in a Portfolio with a site and a semantic name.

CBL_isAsIntended(). Method in class com.adobe.cbl. PFElementWrapper
Test this element to see if it has the intended semantic name.

Elements are normally acquired by doing a lookup in a Portfolio with a site and a semantic name.

CBL_isStructuredData(). Method in interface com.adobe.cbl. PFElement
Check if the data is structured or unstructured.

CBL_isStructuredData(). Method in class com.adobe.cbl. PFElementWrapper

CBL_listProps(). Method in interface com.adobe.cbl. Portfolio
List properties.

CBL_negotiateSemanticName(PFElement, Vector). Method in interface com.adobe.cbl. Portfolio
Negotiate new semantic name for element.

().Methodininterfacecom.adobe.cbl. Portfolio
 Createanemptynewgroupofsites.

(SiteEnumeration).Methodininterfacecom.adobe.cbl. Portfolio
 Createanewgroupofsites.

CBL_newSite().Methodininterfacecom.adobe.cbl. PFElement
 Createanewbindingsiteonthiselementfromscratch.

 ThePortfoliowillgensymannewUIDfortheSite,andusetheRIDthatthePortfoliowascreated
 with.

CBL_newSite().Methodinclasscom.adobe.cbl. PFElementWrapper
 CBL_newSite

 (Object,RectPro,Site).Methodininterfacecom.adobe.cbl. PFElement
 Create/imposebindingsiteongraphicsub-content.

 Usethisonlyoneelementwhosepreferredsemanticnamecomesfromthe GfxCModeloroneof
 itssubinterfaces.

(Object,RectPro,Site).Methodinclasscom.adobe.cbl. PFElementWrapper

CBL_newSubStyleSite(Object,String,Site).Methodininterfacecom.adobe.cbl. PFElement
 Create/imposebindingsiteonasub-style.

 Usethisonlyoneelementwhosepreferredsemanticnamecomesfromthe StylesLModeloroneof
 itssubinterfaces.

CBL_newSubStyleSite(Object,String,Site).Methodinclasscom.adobe.cbl. PFElementWrapper

CBL_newSubTxtSite(Object,int,int,Site).Methodininterfacecom.adobe.cbl. PFElement
 Create/imposebindingsiteontextsub-content.

 Usethisonlyoneelementwhosepreferredsemanticnamecomesfromthe TxtCModeloroneof
 itssubinterfaces.

CBL_newSubTxtSite(Object,int,int,Site).Methodinclasscom.adobe.cbl. PFElementWrapper

CBL_newSubTxtSite(Object,int,Site).Methodininterfacecom.adobe.cbl. PFElement
 Create/imposebindingsitetoendoftextsub-content.

Use this only on elements whose preferred semantic name comes from the TxtCModel or one of its subinterfaces.

CBL_newSubTxtSite(Object,int,Site).Method in class com.adobe.cbl. PFElementWrapper

CBL_open().Method in interface com.adobe.cbl. Portfolio

Open the portfolio.

Can be used for underlying file management, DB session and locking, etc.

CBL_portfolioAttributes().Method in interface com.adobe.cbl. Portfolio

Enumerate all of the attribute names in this portfolio.

CBL_restrictModel(Site,String).Method in interface com.adobe.cbl. Portfolio

Set a model binding.

A Portfolio instance can restrict the model (default semantic name) for a given Site.

CBL_setElementAttr(String,Object,boolean).Method in interface com.adobe.cbl. PFElement

Set an arbitrary attribute on the element.

CBL_setElementAttr(String,Object,boolean).Method in class com.adobe.cbl. PFElementWrapper

CBL_setNickname(String).Method in interface com.adobe.cbl. PFElement

Set an informal description.

IMPLEMENTATION NOTE: If your implementation supports the general attribute methods listed above, you are encouraged to implement informal descriptions with an attribute named "description.informal" (you can use the **NICKNAME** constant defined above.)

CBL_setNickname(String).Method in class com.adobe.cbl. PFElementWrapper

CBL_setPortfolioAttr(String,Object).Method in interface com.adobe.cbl. Portfolio

Set an attribute on this portfolio, with default retention (true).

Same as

`CBL_setPortfolioAttr(attr,val,true)`

CBL_setPortfolioAttr(String,Object,boolean).Method in interface com.adobe.cbl. Portfolio

Set an attribute on this portfolio with retain flag.

CBL_setProp(String, String). Method in interface com.adobe.cbl.Portfolio
Set a property.

(boolean). Method in interface com.adobe.cbl. DirectBinding
Set binding requirement.

(boolean). Method in class com.adobe.cbl. PlacementBinding
Set binding requirement.

(boolean). Method in class com.adobe.cbl. StyleBinding
Set binding requirement.

CBLException(String). Constructor for class com.adobe.cbl. CBLException
Basic exception.

.Static variable in interface com.adobe.cbl. Measure
Centimeters.

CHANNEL. Static variable in interface com.adobe.cbl. AudioDynCModel
Identifies a time-based sequence.

CHANNEL. Static variable in interface com.adobe.cbl. RasterGfxCModel
Identifies a channel.

CHAR. Static variable in interface com.adobe.cbl. TxtStylesLModel
Identifies a character style.

CHARMAP. Static variable in interface com.adobe.cbl. FontStylesLModel
Identifies a character mapping

checkConformance(Class, PFElement, int). Static method in class com.adobe.cbl. ModelHelper
Test an element against the namespace of a model.

checkName(Class, String). Static method in class com.adobe.cbl. ModelHelper
Test a full semantic name against the namespace of a model.

checkTargets(). Method in class com.adobe.cbl. StyleBinding
Get interpretation of target sites.
See description of return values.

CLIP. Static variable in interface com.adobe.cbl.DynCModel

Identifies a time-based sequence.

clone(). Method in class com.adobe.cbl. BindingSpec

Override method for clone.

clone(). Method in class com.adobe.cbl. CompSequence

Override method for clone.

COLOR. Static variable in interface com.adobe.cbl. ColorStylesLModel

Identifies a device independent color.

CommitFailed_CBLErrorException(String). Constructor for class

com.adobe.cbl.CommitFailed_CBLErrorException

Constructs an exception for failure to open.

CompBlock(). Constructor for class com.adobe.cbl. CompBlock

Construct an empty block.

CompBlock(CompSequence). Constructor for class com.adobe.cbl. CompBlock

Construct a block with parent.

CompSequence(). Constructor for class com.adobe.cbl. CompSequence

Construct an empty composition sequence with no templates.

CompSequence(Enumeration). Constructor for class com.adobe.cbl. CompSequence

Construct a composition sequence with blocks and no templates.

CompSequence(Enumeration, Enumeration). Constructor for class com.adobe.cbl. CompSequence

Construct a composition sequence with blocks and templates.

CONTAINER. Static variable in interface com.adobe.cbl. PlacementLModel

Identifies a container.

CONTENT. Static variable in interface com.adobe.cbl. CModel

Identifies all sub-interfaces as content models

CONTENT. Static variable in class com.adobe.cbl. PortfolioType

Content type.

Corner(String). Constructor for class com.adobe.cbl.Corner

CURVE.Staticvariableininterfacecom.adobe.cbl. VectorGfxCModel
Identifiesacurve.

D

DATA.Staticvariableinclasscom.adobe.cbl. SiteUsage
Associatedwithsub-content.

deepClone(Vector).Staticmethodinclasscom.adobe.cbl. BindingSpec
Vectorcloningconveniencemethod.

DEVICE.Staticvariableininterfacecom.adobe.cbl. ColorStylesLModel
Identifiesadevicedependentcolor.

DFLT.Staticvariableininterfacecom.adobe.cbl. PFElement
Defaultsemanticname.

DimensionPro(double,double,Measure.Unit).Constructorforclasscom.adobe.cbl. DimensionPro
Constructadimension.

DIRECTION.Staticvariableininterfacecom.adobe.cbl. DynStylesLModel
Identifiesaplaybackdirection.

DocSpec().Constructorforclasscom.adobe.cbl. DocSpec
CreateanemptyDocSpec

DocSpec(ContentPortfolio,LayoutPortfolio,BindingSpec).Constructorforclass
com.adobe.cbl.DocSpec
CreateacompleteDocSpec.

dummy().Methodinclasscom.adobe.cbl. PFElementWrapper
Needamethodhereforthesectioncomment.

E

eAttributes(PFElement).Methodinclasscom.adobe.cbl. APIHelper
ListPFElementattributes.

eBindingSites(PFElement).Methodinclasscom.adobe.cbl. APIHelper
ListPFElementbindingsites.

eData(PFElement).Methodinclasscom.adobe.cbl. APIHelper
ListPFElementdata.

ELEMENT.Staticvariableinclasscom.adobe.cbl. SiteUsage
Associatedwithanelement.

ElementBinding(String,Vector).Constructorforclasscom.adobe.cbl. ElementBinding
Constructanelementbinding.

END.Staticvariableininterfacecom.adobe.cbl. DynamicsLModel
Identifiesandendtime.

Env()..Constructorforclasscom.adobe.cbl. Env
Constructanemptyenvironment.

equals(Object).Methodinclasscom.adobe.cbl. ElementBinding
Overridetheequals()methodforspecialprocessing.

equals(Object).Methodinclasscom.adobe.cbl. Site
Testforequality.

equals(Object).Methodinclasscom.adobe.cbl. TemplateUsage
Equalityisbasedontheidentifierstring.

eSemName(PFElement).Methodinclasscom.adobe.cbl. APIHelper
ListPFElementdefaultsemanticname.

eSubSites(PFElement).Methodinclasscom.adobe.cbl. APIHelper
ListPFElementsub-sites.

EVENT.Staticvariableininterfacecom.adobe.cbl. DynamicsLModel
Identifiesanevent.

EXPLICIT. Static variable in interface com.adobe.cbl.PlacementLModel
Identifies explicit placement.

FIGURE. Static variable in interface com.adobe.cbl. GfxCModel
Identifies graphic contents.

FILL. Static variable in interface com.adobe.cbl. GfxStylesLModel
Identifies a fill style.

FIRST. Static variable in class com.adobe.cbl. TemplateUsage
First (page) template.

FLOW. Static variable in interface com.adobe.cbl. PlacementLModel
Identifies a flow.

FONT. Static variable in interface com.adobe.cbl. FontStylesLModel
Identifies a font

FOOTNOTE. Static variable in interface com.adobe.cbl. FlowTxtCModel
A footnote.

getArray(). Method in interface com.adobe.cbl. SiteEnumeration
Get the site enumeration as an array.

Always return the full list of sites, even if nextElement() has already been called on this instance.

getArray(). Method in class com.adobe.cbl. VSiteEnumeration
Get the site enumeration as an array.

Always return the full list of sites, even if nextElement() has already been called on this instance.

getBindingSpec(). Method in class com.adobe.cbl. DocSpec
Get the binding specification.

getCompBlocks(). Method in class com.adobe.cbl.CompSequence

Getcompositionblocks.

Ifmorethanoneblockisreturned,andeachblockhasaplacementbinding,allplacementsshould happeninparallelinthecurrenttemplatecontainer.

getCompSeqs(). Method in class com.adobe.cbl. BindingSpec

Getalistofallthecompositionsequencesinorder.

getContentPortfolio(). Method in class com.adobe.cbl. DocSpec

Getthecontentportfolio.

getCorner(). Method in class com.adobe.cbl. RectPro

Getthelocationofthecorneroftherectangle.

getSemanticName(). Method in class com.adobe.cbl. ElementBinding

Getthesemanticname.

getDfltSites(). Method in class com.adobe.cbl. ElementBinding

Getthebindingsitesforthelayoutelementsassociatedwiththiselement.

getDirectBindings(Site). Method in class com.adobe.cbl. CompBlock

Getdirectbindings.

Directbindingsmaybeplacementbindings,stylebindings,etc.

getElementBindings(). Method in class com.adobe.cbl. BindingSpec

Getalistofalloftheelementbindings.

TheelementsoftheenumerationareElementBindingobjects.

getEnum(). Static method in class com.adobe.cbl. Corner

GetanEnumeration.

getEnum(). Static method in class com.adobe.cbl. PortfolioType

GetanEnumeration.

getEnum(). Static method in class com.adobe.cbl. SiteUsage

GetanEnumeration.

getEnum(). Static method in class com.adobe.cbl. TemplateUsage

Get an Enumeration.

getLayoutPortfolio().Methodinclasscom.adobe.cbl. DocSpec

Getthelayoutportfolio.

getModelBindings().Methodinclasscom.adobe.cbl. BindingSpec

Getalistofbindingswhichgloballyconstrainsitetoaparticularintendedmodel.

TheelementsoftheenumerationareModelBindingobjects.

getNamespace(Class).Staticmethodinclasscom.adobe.cbl. ModelHelper

Returnsanarrayofthesemanticnamesimplementedbythismodel.

getNestingLevel().Methodinclasscom.adobe.cbl. APIHelper

Getthenestinglevel.

getOrderedStyles().Methodinclasscom.adobe.cbl. CompBlock

Getorderedstyles.

().Methodinclasscom.adobe.cbl. RectPro

Getthelocationoftheoriginofthecoordinatesystem.

getRestrictedSemanticName().Methodinclasscom.adobe.cbl. ModelBinding

GetthesemanticnamethatrestrictstheSite.

ThetypeofPortfoliothattheSiteisexpectedtobefoundinisimpliedbythemodelofthesemanticname,i.e.,ContentPorfolioforCModelandLayoutPortfolioforLModel.

getRestrictedSite().Methodinclasscom.adobe.cbl. ModelBinding

GettheSitetoberestricted.

getRID().Methodinclasscom.adobe.cbl. BindingSpec

ReturntheconstantrefereridforthisBindingSpec.

ImplementationfortheRefererinterface.

getRID().Methodininterfacecom.adobe.cbl. Referer

Returntherefereridentifier.

NOTE:thismethodshouldalwaysreturnthesamevalueforanygiveninstancethatimplements thisinterface.

getTemplates().Methodinclasscom.adobe.cbl. CompSequence

Get templates.

getUnit().Methodinclasscom.adobe.cbl. DimensionPro
Gettheunitofmeasure.

getUnit().Methodinclasscom.adobe.cbl. IntervalPro
Gettheunitofmeasure.

getUnit().Methodinclasscom.adobe.cbl. RectPro
Gettheunitofmeasure.

.Staticvariableinclasscom.adobe.cbl. SiteUsage
Associatedwithagroup.

H

hashCode().Methodinclasscom.adobe.cbl. Site
Supplyanefficienthashingfunction.

hasMoreElements().Methodinclasscom.adobe.cbl. PFEnumeration
Moreelements?

hasMoreElements().Methodininterfacecom.adobe.cbl. SiteEnumeration
Moreelements?

hasMoreElements().Methodinclasscom.adobe.cbl. VSiteEnumeration
Moreelements?

.Staticvariableininterfacecom.adobe.cbl. FlowTxtCModel
Aheadingforanarticle,section,orlist.

height.Variableinclasscom.adobe.cbl. DimensionPro

height.Variableinclasscom.adobe.cbl. RectPro

INCH.Staticvariableininterfacecom.adobe.cbl. Measure
Inches.

indent().Methodinclasscom.adobe.cbl. APIHelper
Indentbylevel.

IntervalPro(String,String,Measure.Unit).Constructorforclasscom.adobe.cbl. IntervalPro
Constructaninterval.

isValidSite(PFElement,Site).Staticmethodinclasscom.adobe.cbl. PFElementWrapper
Utilitytocheckifabindingsiteispresentonanelement.

LABEL.Staticvariableininterfacecom.adobe.cbl. FlowTxtCModel
Atextlabel.

LAST.Staticvariableinclasscom.adobe.cbl. TemplateUsage
Last(page)template.

LAYER.Staticvariableininterfacecom.adobe.cbl. LayeredGfxCModel
Identifiesalayer.

LAYOUT.Staticvariableininterfacecom.adobe.cbl. LModel
Identifiesallsub-interfacesaslayoutmodels

LAYOUT.Staticvariableinclasscom.adobe.cbl. PortfolioType
Layouttype.

LEFT.Staticvariableinclasscom.adobe.cbl. TemplateUsage
Left-handtemplate.

level.Variableinclasscom.adobe.cbl. APIHelper

LIGHT.Staticvariableininterfacecom.adobe.cbl. TriDGfxCModel
Identifiesalightsource.

limit. Variable in class com.adobe.cbl.LimitCheckException

LimitCheckException(String,int,String,int).Constructorforclass
com.adobe.cbl.LimitCheckException

Constructsanexceptionforanumthatexceedsalimit.

LINE.Staticvariableininterfacecom.adobe.cbl. GfxStylesLModel

Identifiesalifestyle.

LINE.Staticvariableininterfacecom.adobe.cbl. VectorGfxCModel

Identifiesaline.

LINK.Staticvariableininterfacecom.adobe.cbl. LinkCModel

Identifiesalink

LIST.Staticvariableininterfacecom.adobe.cbl. FlowTxtCModel

Alistofitems,withanoptionalheading.

.Staticvariableininterfacecom.adobe.cbl. FlowTxtCModel

Aniteminalist.

LOCATION.Staticvariableininterfacecom.adobe.cbl. PlacementLModel

Identifiesalocation.

M

ModelBinding(String,Site).Constructorforclasscom.adobe.cbl. ModelBinding

Constructarestrictedmodelbinding.

()..Constructorforclasscom.adobe.cbl. ModelHelper

msg.Variableinclasscom.adobe.cbl. LimitCheckException

Instancevariables

N

nestIn().Methodinclasscom.adobe.cbl. APIHelper
Nestoneleveldown.

nestOut().Methodinclasscom.adobe.cbl. APIHelper
Nestonelevelup.

nextElement().Methodinclasscom.adobe.cbl. PFEnumeration
Getthenextelement.

nextElement().Methodininterfacecom.adobe.cbl. SiteEnumeration
Getthenextelement.

nextElement().Methodinclasscom.adobe.cbl. VSiteEnumeration
Getthenextelement.

nextSite().Methodininterfacecom.adobe.cbl. SiteEnumeration
GetthenextSite.

nextSite().Methodinclasscom.adobe.cbl. VSiteEnumeration
GetthenextSite.

.Staticvariableininterfacecom.adobe.cbl. PFElement
Attributenamefornickname.

num.Variableinclasscom.adobe.cbl. LimitCheckException

OpenFailed_CBLErrorException(String).Constructorforclasscom.adobe.cbl. OpenFailed_CBLErrorException
Constructsanexceptionforfailurestoopen.

orig.Variableinclasscom.adobe.cbl. PFElementWrapper
Thetrue,originalelementinthePortfolio.

out.Variableinclasscom.adobe.cbl. APIHelper

OVER. Static variable in class com.adobe.cbl.LimitCheckException

Upper limit exceeded.

OVERLAY.Staticvariableininterfacecom.adobe.cbl. DynCModel
Identifiesanoverlay.

P

PAGE.Staticvariableininterfacecom.adobe.cbl. DocGfxCModel
Identifiesapage.

PARA.Staticvariableininterfacecom.adobe.cbl. FlowTxtCModel
Aunitoftext.

PARA.Staticvariableininterfacecom.adobe.cbl. TxtStylesLModel
Identifiesaparagraphstyle.

parseElement(String).Staticmethodinclasscom.adobe.cbl. ModelHelper
Extracttheelementnamefromthefullsemanticname.

parseModel(String).Staticmethodinclasscom.adobe.cbl. ModelHelper
Extractthemodelfromthefullsemanticname.

PATH.Staticvariableininterfacecom.adobe.cbl. VectorGfxCModel
Identifiesapath.

pfAttributes(Portfolio).Methodinclasscom.adobe.cbl. APIHelper
ListPortfolioattributes.

pfElements(Portfolio).Methodinclasscom.adobe.cbl. APIHelper
ListPortfolioelements.

PFElementWrapper(PFElement,Site).Constructorforclasscom.adobe.cbl. PFElementWrapper
Constructwithpreferredname.

PFElementWrapper(PFElement,Site,String,boolean).Constructorforclass
com.adobe.cbl.PFElementWrapper
Constructfromscratch.

PFEnumeration(Portfolio).Constructorforclasscom.adobe.cbl. PFEnumeration

pfGlobals(Portfolio). Method in class com.adobe.cbl.APIHelper
ListPortfolioglobalinformation.

pfGroups(Portfolio). Method in class com.adobe.cbl. APIHelper
ListPortfoliogroups.

pfModelRestrictions(Portfolio). Method in class com.adobe.cbl. APIHelper
ListModelRestrictions.

pfProperties(Portfolio). Method in class com.adobe.cbl. APIHelper
ListPortfolioproperties.

pfReferer(Portfolio). Method in class com.adobe.cbl. APIHelper
ListReferer.

PIXEL. Static variable in interface com.adobe.cbl. Measure
Pictureelements.

(SiteEnumeration,SiteEnumeration). Constructor for class
com.adobe.cbl.PlacementBinding
Constructarequiredplacementbinding.

(SiteEnumeration,SiteEnumeration,boolean). Constructor for class
com.adobe.cbl.PlacementBinding
Constructaplacementbinding,specifyingrequirement.

pn(String). Method in class com.adobe.cbl. APIHelper
Printwithanewline,observinglevel.

(Site). Method in class com.adobe.cbl. APIHelper
Listabindingsite.

.Staticvariableininterfacecom.adobe.cbl. Measure
Printer'spoint:72.27pi.

.Staticvariableininterfacecom.adobe.cbl. VectorGfxCModel
Identifiesapoint.

POLYGON. Static variable in interface com.adobe.cbl. TriDGfxCModel
Identifiesapolygonin3-space.

PORTFOLIO. Static variable in interface com.adobe.cbl.Model
SemanticnameforaPortfolio.

PortfolioType(String). Constructor for class com.adobe.cbl. PortfolioType
.Staticvariableininterfacecom.adobe.cbl. PlacedTxtCModel
Preformattedtext.

PREF. Static variable in interface com.adobe.cbl. PFElement
Preferredsemanticname.

PREVIEW. Static variable in interface com.adobe.cbl. MultiGfxCModel
Identifiesapreview.

pri(String). Method in class com.adobe.cbl. APIHelper
Printwithoutnewline,ignoringlevel.

.Staticvariableininterfacecom.adobe.cbl. Measure
PostScriptpoint:72pi

R

RASTER. Static variable in interface com.adobe.cbl. RasterGfxCModel
Identifiesarastergraphic.

.Staticvariableininterfacecom.adobe.cbl. DynStylesLModel
Identifiesaplaybackrate.

RectPro(double,double,double,double,Measure.Unit,Corner,Corner). Constructor for class
com.adobe.cbl.RectPro
Constructarectangle.

Therectangleisrequiredtobewell-formed.

REFERENCE. Static variable in interface com.adobe.cbl. Model
SemanticnamefortheanindirectreferencetoaPFElementinanotherPortfolio.

removeCompBlock(CompBlock). Method in class com.adobe.cbl. CompSequence
Removeacompositionblock.

removeCompSeq(CompSequence). Method in class com.adobe.cbl.BindingSpec

Removeacompositionsequence.

removeDirectBinding(DirectBinding).Methodinclasscom.adobe.cbl. CompBlock

Removedirectbinding.

removeElementBinding(ElementBinding).Methodinclasscom.adobe.cbl. BindingSpec

Removeelementbinding.

removeModelBinding(ModelBinding).Methodinclasscom.adobe.cbl. BindingSpec

Removemodelbinding.

removeOrderedStyles(CompBlock).Methodinclasscom.adobe.cbl. CompBlock

Removeorderedstyles.

resetEnumeration().Methodinclasscom.adobe.cbl. PFEnumeration

Resetenumeration.

Oncetheenumerationisexhausted,thisobjectwillcontinuetothrowNoSuchElementException
untilitisreset.

resetEnumeration().Methodinclasscom.adobe.cbl. VSiteEnumeration

Resetenumeration.

Oncetheenumerationisexhausted,thisobjectwillcontinuetothrowNoSuchElementException
untilitisreset.

rid.Variableinclasscom.adobe.cbl. Site

TheRefereridentifier.

.Staticvariableinclasscom.adobe.cbl. TemplateUsage

Right-handtemplate.

S

SAMPLE.Staticvariableininterfacecom.adobe.cbl. Measure

Digitalsample;audio,etc.

SECTION.Staticvariableininterfacecom.adobe.cbl. FlowTxtCModel

A logical division, with an optional heading.

setBindingSpec(BindingSpec).Methodinclasscom.adobe.cbl. DocSpec
Setthebindingspecificationportfolio.

setContentPortfolio(ContentPortfolio).Methodinclasscom.adobe.cbl. DocSpec
Setthecontentportfolio.

setLayoutPortfolio(LayoutPortfolio).Methodinclasscom.adobe.cbl. DocSpec
Setthelayoutportfolio.

setNestingLevel(int).Methodinclasscom.adobe.cbl. APIHelper
Setthenestinglevel.

setRID(String).Methodinclasscom.adobe.cbl. BindingSpec
SettheconstantrefereridforthisBindingSpec.
ThecurrentsettingfortheRIDisrequiredbenull(i.e.,getRID()mustreturnnull).

setTemplates(Enumeration).Methodinclasscom.adobe.cbl. CompSequence
Settemplates.
Replacestheexistingtemplates.

Site(String).Constructorforclasscom.adobe.cbl. Site
CreateaSitefromacompositeString.
Giventhefollowingsetup:

```
SiteaSite=newSite(rid,uid);  
SitebSite=newSite(aSite.toString());
```

Arrangeforthefollowingexpressiontobetrue:

```
aSite.equals(bSite)&&bSite.equals(aSite)
```

Site(String,String).Constructorforclasscom.adobe.cbl. Site
CreateaSite.

sites.Variableinclasscom.adobe.cbl. VSiteEnumeration
Thestoragevector.

SiteUsage(String). Constructor for class com.adobe.cbl.SiteUsage

START.Staticvariableininterfacecom.adobe.cbl. DynamicsLModel
Identifiesastarttime.

STYLE.Staticvariableininterfacecom.adobe.cbl. StylesLModel
Identifiesastyle.

StyleBinding(SiteEnumeration,SiteEnumeration).Constructorforclasscom.adobe.cbl. StyleBinding
Constructarequiredstylebindingforcontent.

StyleBinding(SiteEnumeration,SiteEnumeration,boolean).Constructorforclass
com.adobe.cbl.StyleBinding
Constructarequiredstylebindingwithdesignatedtargets.

StyleBinding(SiteEnumeration,SiteEnumeration,boolean,boolean).Constructorforclass
com.adobe.cbl.StyleBinding
Constructastylebindingwithdesignatedtargets,indicatingrequirement.

.Variableinclasscom.adobe.cbl. APIHelper

TCODE.Staticvariableininterfacecom.adobe.cbl. TxtCModel
Identifiestextencodeddata,suchasaJavaScript

TEMPLATE.Staticvariableininterfacecom.adobe.cbl. PlacementLModel
Identifiesatemplate.

.Variableinclasscom.adobe.cbl. Template
Bindingsitefortemplateelement.

Template(Site,Enumeration).Constructorforclasscom.adobe.cbl. Template
Constructwithatemplateandusagelist.

TemplateUsage(String).Constructorforclasscom.adobe.cbl. TemplateUsage
Constructthenextelementoftheenumeration.

TEXT.Staticvariableininterfacecom.adobe.cbl. TxtCModel

Identifies any kind of human readable text content

TEXTURE.Staticvariableininterfacecom.adobe.cbl. GfxCModel
Identifiesatexture.

TIMEBASE.Staticvariableininterfacecom.adobe.cbl. DynCModel
Identifiesatime-base.

TIMER.Staticvariableininterfacecom.adobe.cbl. DynamicsLModel
Identifiesatimer.

TOP_LEFT.Staticvariableinclasscom.adobe.cbl. Corner
xincreasestotheright,yincreasesdown.

TOP_RIGHT.Staticvariableinclasscom.adobe.cbl. Corner
xincreasestothelleft,yincreasesdown.

toString().Methodinclasscom.adobe.cbl. Corner
Returnthedeclaredidentifierstring.

toString().Methodinclasscom.adobe.cbl. PortfolioType
Returnthedeclaredidentifierstring.

toString().Methodinclasscom.adobe.cbl. Site
SupplyaStringvalueforthisobject.

toString().Methodinclasscom.adobe.cbl. SiteUsage
Returnthedeclaredidentifierstring.

toString().Methodinclasscom.adobe.cbl. TemplateUsage
Returnthedeclaredidentifierstring.

TRACK.Staticvariableininterfacecom.adobe.cbl. VideoDynCModel
Identifiesavideotrack.

TRIDMODEL.Staticvariableininterfacecom.adobe.cbl. TriDGfxCModel
Identifies3Dmodel.

U

uid. Variable in class com.adobe.cbl. Site

The unique reference identifier.

UID_FORBIDDEN. Static variable in class com.adobe.cbl. Site

Unique reference identifiers are forbidden from containing this character, in the form of a single character String.

.Static variable in class com.adobe.cbl. Site

Unique reference identifiers are forbidden from containing this character.

UNBOUND. Static variable in interface com.adobe.cbl. CModel

Bindings with no content, used for Templates.

UNDER. Static variable in class com.adobe.cbl. LimitCheckException

Lower limit exceeded.

unimplemented(). Static method in class com.adobe.cbl. CBLEException

Throw a RuntimeException for unimplemented methods.

UNKNOWN. Static variable in interface com.adobe.cbl. Model

Name for an element of unknown semantics.

UnsupportedFeature_CBLEException(String). Constructor for class

com.adobe.cbl.UnsupportedFeature_CBLEException

Constructs an exception for unsupported features.

usage. Variable in class com.adobe.cbl. Template

Optional list of usages.

V

VERTEX. Static variable in interface com.adobe.cbl. TriDGfxCModel

Identifies a vertex.

violation. Variable in class com.adobe.cbl. LimitCheckException

(Vector). Constructor for class com.adobe.cbl.VSiteEnumeration

Construct with a Vector.

.Variableinclasscom.adobe.cbl. DimensionPro

.Variableinclasscom.adobe.cbl. RectPro

WIREFRAME.Staticvariableininterfacecom.adobe.cbl. MultiGfxCModel

Identifiesawireframeview.

writePFElement(PFElement,APIHelper,boolean).Staticmethodinclasscom.adobe.cbl. APIHelper

ListthecontentsofaPFElement.

Writeouttheattributes,children,bindingsites,anddataofaPFElementinacanonicalorder.

writePortfolio(Portfolio,APIHelper).Staticmethodinclasscom.adobe.cbl. APIHelper

ListthecontentsofaPortfolio.

WriteoutcontentsofaPortfolioinacanonicalorder.

X

.Variableinclasscom.adobe.cbl. RectPro

Y

.Variableinclasscom.adobe.cbl. RectPro

Class com.adobe.cbl.APIHelper

```
java.lang.Object
|
+----com.adobe.cbl.APIHelper
```

public class **APIHelper**

extends Object

APIHelper helps debug/output Portfolio and PFElement objects.

Field Index

- level
- out
- tab

Constructor Index

- **APIHelper**(Writer)
Construct with a Writer.

Method Index

- **eAttributes**(PFElement)
List PFElement attributes.

- **eBindingSites(PFElement)**
ListPFElementbindingsites.
- **eData(PFElement)**
ListPFElementdata.
- **eSemName(PFElement)**
ListPFElementdefaultsemanticname.
- **eSubSites(PFElement)**
ListPFElementsub-sites.
- **getNestingLevel()**
Getthenestinglevel.
- **indent()**
Indentbylevel.
- **nestIn()**
Nestoneleveldown.
- **nestOut()**
Nestonelevelup.
- **pfAttributes(Portfolio)**
ListPortfolioattributes.
- **pfElements(Portfolio)**
ListPortfolioelements.
- **pfGlobals(Portfolio)**
ListPortfolioglobalinformation.
- **pfGroups(Portfolio)**
ListPortfoliogroups.
- **pfModelRestrictions(Portfolio)**
ListModelRestrictions.
- **pfProperties(Portfolio)**
List Portfolio properties.

- **pfReferer(Portfolio)**

ListReferer.

- **pn(String)**

Printwithanewline,observinglevel.

- **pnBindingSite(Site)**

Listbindingsite.

- **pri(String)**

Printwithoutnewline,ignoringlevel.

- **setNestingLevel(int)**

Setthenestinglevel.

- **writePFELEMENT(PFElement,APIHelper,boolean)**

ListthecontentsofaPFElement.

Writeouttheattributes,children,bindingsites,anddataofaPFElementinacanonicalorder.

- **writePortfolio(Portfolio,APIHelper)**

ListthecontentsofaPortfolio.

WriteoutcontentsofaPortfolioinacanonicalorder.

Fields

- **tab**

```
public String tab
```

- **out**

```
protected PrintWriter out
```

- **level**

```
protected int level
```

Constructors

● APIHelper

```
public APIHelper(Writer out)
```

Construct with a Writer.

Methods

● writePortfolio

```
public static void writePortfolio(Portfolio p,  
                                  APIHelper h)
```

List the contents of a Portfolio.

Write out contents of a Portfolio in a canonical order.

Parameters:

p- The Portfolio to list.

h- The APIHelper to use. Could be a subclass of this class to achieve a different format.

Assumes that nesting level has already been set.

● writePFElement

```
public static void writePFElement(PFElement anElem,  
                                   APIHelper h,  
                                   boolean deep)
```

List the contents of a PFElement.

Write out the attributes, children, binding sites, and data of a PFElement in a canonical order.

Parameters:

anElem- The PFElement to list.

h- The APIHelper to use. Could be a subclass of this class to achieve a different format.

Assumes that nesting level has already been set.

deep - True to enumerate all children, false to write this element only.

● **setNestingLevel**

```
public void setNestingLevel(int level)
```

Setthenestinglevel.

● **getNestingLevel**

```
public int getNestingLevel()
```

Getthenestinglevel.

● **nestIn**

```
public void nestIn()
```

Nestoneleveldown.

● **nestOut**

```
public void nestOut()
```

Nestonelevelup.

● **pn**

```
public void pn(String msg)
```

Printwithanewline,observinglevel.

● **pri**

```
public void pri(String msg)
```

Printwithoutnewline,ignoringlevel.

● **indent**

```
public void indent()
```

```
    Indentbylevel.
```

● **pfGlobals**

```
public void pfGlobals(Portfoliop)
```

```
    ListPortfolioglobalinformation.
```

● **pfProperties**

```
public void pfProperties(Portfoliop)
```

```
    ListPortfolioproperties.
```

● **pfAttributes**

```
public void pfAttributes(Portfoliop)
```

```
    ListPortfolioattributes.
```

● **pfElements**

```
public void pfElements(Portfoliop)
```

```
    ListPortfolioelements.
```

● **pfGroups**

```
public void pfGroups(Portfoliop)
```

```
    ListPortfoliogroups.
```

● **pfModelRestrictions**

```
public void pfModelRestrictions(Portfoliop)
```

```
    ListModelRestrictions.
```

● pfReferer

```
public void pfReferer(Portfoliop)
    ListReferer.
```

● eSemName

```
public void eSemName(PFElemente)
    ListPFElementdefaultsemanticname.
```

● eAttributes

```
public void eAttributes(PFElemente)
    ListPFElementattributes.
```

● eBindingSites

```
public void eBindingSites(PFElemente)
    ListPFElementbindingsites.
```

● eData

```
public void eData(PFElemente)
    ListPFElementdata.
```

● eSubSites

```
public void eSubSites(PFElemente)
    ListPFElementsub-sites.
```

● pnBindingSite

```
public void pnBindingSite(Site aSite)
```

Listabindingsite.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Interface

com.adobe.cbl.AudioDynCModel

AudioDynCModel

extends `DynCModel`

AudioDynCModel defines audio content.

See Also:

`CModel`

Field Index

• CHANNEL

Identifies a time-based sequence.

Fields

• CHANNEL

```
public static final String CHANNEL
```

Identifies a time-based sequence.

Interface com.adobe.cbl.DynCModel

DynCModel

extends CModel

DynCModel defines dynamic media content.

See Also:

CModel

Field Index

• CLIP

Identifies a time-based sequence.

• OVERLAY

Identifies an overlay.

• TIMEBASE

Identifies a time-base.

Fields

● CLIP

Identifies a time-based sequence.

● TIMEBASE

public static final String TIMEBASE

Identifiesatime-base.

● OVERLAY

public static final String OVERLAY

Identifiesanoverlay.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Class com.adobe.cbl.BindingSpec

```
java.lang.Object
|
+----com.adobe.cbl.BindingSpec
```

public class **BindingSpec**

extends Object

implements Referer,Cloneable

BindingSpec specifies document bindings.

to a particular content and/or layout element. See the CBL API User's Guide for further details.

Constructor Index

• **BindingSpec()**

Create an empty BindingSpec.

• **BindingSpec(String)**

Create an empty BindingSpec with RID.

Method Index

• **addAfter(Vector, Object, Object)**

Vector insert convenience method.

• **addCompSeq(CompSequence, CompSequence)**

Add a direct binding composition sequence in order.

- **addElementBinding(ElementBinding)**

Add an element binding.

Order is not maintained.

- **addModelBinding(ModelBinding)**

Add a model binding.

Order is not maintained.

- **clone()**

Override method for clone.

- **deepClone(Vector)**

Vector cloning convenience method.

- **getCompSeqs()**

Get a list of all the composition sequences in order.

- **getElementBindings()**

Get a list of all of the element bindings.

The elements of the enumeration are ElementBinding objects.

- **getModelBindings()**

Get a list of bindings which globally constrain a Site to a particular intended model.

The elements of the enumeration are ModelBinding objects.

- **getRID()**

Return the constant referer id for this BindingSpec.

Implementation for the Referer interface.

- **removeCompSeq(CompSequence)**

Remove a composition sequence.

- **removeElementBinding(ElementBinding)**

Remove element binding.

- **removeModelBinding(ModelBinding)**

Remove model binding.

• **setRID(String)**

Set the constant referer id for this BindingSpec.

The current setting for the RID is required to be null (i.e., `getRID()` must return null).

Constructors

• **BindingSpec**

```
public BindingSpec()
```

Create an empty BindingSpec.

• **BindingSpec**

```
public BindingSpec(String rid)
```

Create an empty BindingSpec with RID.

Methods

• **addAfter**

```
public static void addAfter(Vector v,  
                           Object after,  
                           Object elem)
```

Vector insert convenience method.

Parameters:

v- Vector to add to.

after- Element to add after, null for insert at beginning.

elem- Element to add.

• **deepClone**

```
public static Vector deepClone(Vector v)
```

Vector cloning convenience method.

Parameters:

v-Required to be the original Vector (not already shallow cloned.)

Returns:

Deeply cloned vector, including returned object itself.

● **getElementBindings**

```
public Enumeration getElementBindings()
```

Get a list of all of the element bindings.

The elements of the enumeration are ElementBinding objects. May return null.

● **getModelBindings**

```
public Enumeration getModelBindings()
```

Get a list of bindings which globally constrain a Site to a particular intended model.

The elements of the enumeration are ModelBinding objects. May return null.

● **addElementBinding**

```
public void addElementBinding(ElementBinding anElementBinding)
```

Add an element binding.

Order is not maintained. If the ElementBinding matches an existing binding, the old binding is replaced.

Parameters:

anElementBinding-Element binding to add.

● **addModelBinding**

```
public void addModelBinding(ModelBinding aModelBinding)
```

Add a model binding.

Order is not maintained. If the ModelBinding matches an existing binding, the old binding is replaced.

Parameters:

aModelBinding-Model binding to add.

● **removeElementBinding**

```
public void removeElementBinding(ElementBinding aBinding)
```

Remove element binding.

Parameters:

aBinding-The binding to remove. Required to be an ElementBinding currently defined in this BindingSpec.

● **removeModelBinding**

```
ModelBinding aBinding)
```

Remove model binding.

Parameters:

aBinding-The binding to remove. Required to be a ModelBinding currently defined in this BindingSpec.

● **getCompSeqs**

```
public Enumeration getCompSeqs()
```

Get a list of all the composition sequences in order.

Returns:

The elements of the enumeration are CompSequence objects. The enumeration is a copy of the list of bindings: edits made to the bindings list after this method are called are not reflected in the original return value. May return null, meaning that there are no direct bindings in the BindingSpec.

● **addCompSeq**

```
public void addCompSeq(CompSequenceafter,  
                      CompSequenceaSeq)
```

Add a direct binding composition sequence in order.

Parameters:

after- The composition sequence to add after. If null, means insert at beginning. If same object as aSeq, means append to the end.
aSeq- The composition sequence to add.

● **removeCompSeq**

```
public void removeCompSeq(CompSequenceaSeq)
```

Remove a composition sequence.

Parameters:

aSeq- The composition sequence to remove. Required to be a composition sequence in this binding spec.

● **getRID**

```
public String getRID()
```

Return the constant referer id for this BindingSpec.

Implementation for the Referer interface.

Returns:

The constant RID.

● **setRID**

```
protected void setRID(String rid)
```

Set the constant referer id for this BindingSpec.

The current setting for the RID is required to be null (i.e., getRID() must return null).

Parameters:

rid - The RID to set.

● clone

public Object clone() throws CloneNotSupportedException

Override method for clone.

Throws: CloneNotSupportedException

From Object.clone().

Overrides:

clone in class Object

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Interface com.adobe.cbl.Referer

Referer

Referer defines referer identifier.

Method Index

- **getRID()**

Return the referer identifier.

NOTE: this method should always return the same value for any given instance that implements this interface.

Methods

- **getRID**

Return the referer identifier.

NOTE: this method should always return the same value for any given instance that implements this interface.

Returns:

The constant rid.

Interface com.adobe.cbl.CModel

CModel

extends [Model](#)

CModel is the base class at the root of a hierarchy of Universal Models for Content.

Each subclass defines a content model by virtue of its position in the tree. For example:

`CModel -> GfxCModel -> LayeredGfxCModel`

A *LayeredGfxCModel* can be treated as a *GfxCModel* by virtue of its inheritance from that interface. In

CModel subclasses also specify a table of reasonable MIME types? For example, **FlowTxtCModel** might list "text/*". See [PFElement.getData\(\)](#).

See Also:

[Model](#), [GfxCModel](#), [LayeredGfxCModel](#), [FlowTxtCModel](#), [PFElement](#)

Field Index

• CONTENT

Identifies all sub-interfaces as content models

• UNBOUND

Bindings with no content, used for Templates.

Fields

● CONTENT

public static final String CONTENT

Identifies all sub-interfaces a content models

● UNBOUND

public static final String UNBOUND

Bindings with no content, used for Templates.

See Also:

Template

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Model

Model is the base model at the root of a hierarchy of content and layout models.

Model defines a namespace that an author uses to identify layout elements by their semantic interpretation, sometimes called "markup". This namespace is expressed concretely as constants in the interface. There is also a method for acquiring a table (array) of names. Subclasses may extend the

By convention, the string value for semantic names is the same as the name of the constant. This means that the name is composed of the interface name, a dot, and then a string that matches the following regular expression: `[A-Z][A-Z0-9_]*`

Finally, no methods are defined in this interface or any of its derivatives. See the `ModelHelper` class for

See Also:

`ModelHelper`

Field Index

• PORTFOLIO

Semantic name for a Portfolio.

• REFERENCE

Semantic name for the an indirect reference to a PFElement in another Portfolio.

• UNKNOWN

Name for an element of unknown semantics.

Fields

● PORTFOLIO

```
public static final String PORTFOLIO  
    SemanticnameforaPortfolio.
```

● REFERENCE

```
public static final String REFERENCE  
    SemanticnamefortheanindirectreferencetoaPFElementinanotherPortfolio.
```

● UNKNOWN

```
public static final String UNKNOWN  
    Nameforanelementofunknownsemantics.
```

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Interface

com.adobe.cbl.ColorStylesLModel

ColorStylesLModel

extends `StylesLModel`

ColorStylesLModel defines color presentation information.

See Also:

`LModel`

Field Index

• COLOR

Identifies a device independent color.

• DEVICE

Identifies a device dependent color.

Fields

• DEVICE

```
public static final String DEVICE
```

Identifies a device dependent color.

• COLOR

```
public static final String COLOR
```

Identifies a device independent color.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Interface com.adobe.cbl.StylesLModel

StylesLModel

extends LModel

StylesLModel defines presentation information.

See Also:

LModel

Field Index

• STYLE

Identifies a style.

Fields

● STYLE

```
public static final String STYLE
```

Identifies a style.

Class com.adobe.cbl.CompBlock

```
java.lang.Object
|
+----com.adobe.cbl.CompBlock
```

public class **CompBlock**

extends Object

CompBlock specifies a composition block of direct bindings.

Each **CompBlock** defines direct bindings. A **CompBlock** also defines a scope for content placement such that a reference to a particular content binding site may only appear once in every block.

Constructor Index

- **CompBlock()**
Construct an empty block.
- **CompBlock(CompSequence)**
Construct a block with parent.

Method Index

- **addDirectBinding(DirectBinding, DirectBinding)**
Add direct binding.
- **addOrderedStyles(CompBlock, CompBlock)**
Add ordered styles.

- **getDirectBindings(Site)**

Getdirectbindings.

Directbindingsmaybeplacementbindings,stylebindings,etc.

- **getOrderedStyles()**

Getorderedstyles.

- **removeDirectBinding(DirectBinding)**

Removedirectbinding.

- **removeOrderedStyles(CompBlock)**

Removeorderedstyles.

Constructors

- **CompBlock**

```
public CompBlock()
```

Constructanemptyblock.

- **CompBlock**

```
public CompBlock(CompSequenceparent)
```

Constructablockwithparent.

Methods

- **getDirectBindings**

```
public Enumeration getDirectBindings(SiteaSite)
```

Getdirectbindings.

Directbindingsmaybeplacementbindings,stylebindings,etc.Directbindingsareunordered.

Parameters:

aSite-If null, return all direct bindings. If non-null, return only bindings which contain a reference to the specified Site.

Returns:

The elements of the enumeration are DirectBinding objects. The enumeration is a copy of the list of direct bindings: edits made to the list after this method is called will not be reflected in the original return value. May return null, indicating that there are no direct bindings.

● addDirectBinding

```
public void addDirectBinding(DirectBinding after,  
                             DirectBinding aBinding)
```

Add direct binding.

Parameters:

after- Binding after which a binding is added, null for first.
aBinding- The binding to add.

● removeDirectBinding

```
public void removeDirectBinding(DirectBinding aBinding)
```

Remove direct binding.

Parameters:

aBinding- The binding to remove. Required to be a DirectBinding currently defined in this CompBlock.

● getOrderedStyles

```
public Enumeration getOrderedStyles()
```

Get ordered styles.

Returns:

The ordered styles. Each element of the Enumeration is a CompBlock, returned in the specified order. Use CompBlock.getDirectBindings() to enumerate the style bindings in order from the virtual composition block. The Enumeration is a copy of the ordered styles.

contained in this block: edits made to the ordered styles after this method returns will not be reflected in the original return value. Returns null if no ordered styles in this block.

• addOrderedStyles

```
public void addOrderedStyles(CompBlockafter,  
                             CompBlockstyles)
```

Addorderedstyles.

Parameters:

after-CompBlockafterwhichstylesareadded,nullforfirst.
styles-Theorderedstylestoadd.

• removeOrderedStyles

```
public void removeOrderedStyles(CompBlockstyles)
```

Removeorderedstyles.

Parameters:

styles-Thestyletoremove.Requiredtobe aStyleBindingcurrentlydefinedinthis
ordering.

Class com.adobe.cbl.CompSequence

```
java.lang.Object
|
+----com.adobe.cbl.CompSequence
```

public class **CompSequence**

extends Object

implements Cloneable

CompSequence defines a composition sequence with templates.

Each **CompSequence** implicitly means "start a new" instance from the specified list of templates, such as start new page or start new column. The order of **CompBlock** objects is important: it represents the sequence for composition.

Constructor Index

- **CompSequence()**

Construct an empty composition sequence with no templates.

- **CompSequence(Enumeration)**

Construct a composition sequence with blocks and no templates.

- **CompSequence(Enumeration, Enumeration)**

Construct a composition sequence with blocks and templates.

Method Index

- **addCompBlock(CompBlock,CompBlock)**

Addacompositionblock.

- **clone()**

Overridemethodforclone.

- **getCompBlocks()**

Getcompositionblocks.

Ifmorethanoneblockisreturned,andeachblockhasaplacementbinding,allplacementsshould happeninparallelinthecurrenttemplatecontainer.

- **getTemplates()**

Gettemplates.

- **removeCompBlock(CompBlock)**

Removeacompositionblock.

- **setTemplates(Enumeration)**

Settemplates.

Replacetheexistingtemplates.

CONSTRUCTORS

- **CompSequence**

```
public CompSequence()
```

Constructanemptycompositionsequencewithnotemplates.

- **CompSequence**

```
public CompSequence(Enumeration compBlks)
```

Constructacompositionsequencewithblocksandnotemplates.

Parameters:

compBlks-The initial set of blocks. Can be null. Each element is required to be a CompBlock object.

● CompSequence

```
public CompSequence(Enumeration compBlks,  
                    Enumeration templates)
```

Construct a composition sequence with blocks and templates.

Parameters:

compBlks-The initial set of blocks. Can be null.
templates-Each element of the Enumeration is required to be a Template object. Can be null.

Methods

● clone

```
public Object clone() throws CloneNotSupportedException
```

Override method for clone.

Throws: CloneNotSupportedException

From Object.clone().

Overrides:

clone in class Object

● getCompBlocks

```
public Enumeration getCompBlocks()
```

Get composition blocks.

If more than one block is returned, and each block has a placement binding, all placements should happen in parallel in the current template container. For example, if this method returns 2 placement bindings, and the current template is a page of columns, contents should be "poured" as two parallel columns.

Returns:

The elements of the enumeration are `CompBlock` objects. This enumeration is a copy of the list of blocks. Edits to the list of blocks made after this method is called will not be reflected in the original return value. Can return null, meaning no `CompBlocks`.

● addCompBlock

```
public void addCompBlock(CompBlock after,
                        CompBlock aBlock)
```

Add a composition block.

Parameters:

after- Binding after which a block is added, null for first, same object as a block to append at end.
aBlock- The block to add.

● removeCompBlock

```
public void removeCompBlock(CompBlock aBlock)
```

Remove a composition block.

Parameters:

aBlock- The block to remove. Required to be a block in this sequence.

● getTemplates

```
public Enumeration getTemplates()
```

Get templates.

Returns:

Each element in the enumeration is a `Template` object. Can return null, meaning no templates in this sequence.

See Also:

`Template`

● setTemplates

public void setTemplates(Enumeration enumTemps)

Settemplates.

Replacestheexistingtemplates.

Parameters:

enumTemps-EachementintheenumerationisaTemplateobject.Canbenull.

SeeAlso:

Template

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Interface com.adobe.cbl.ContentPortfolio

ContentPortfolio

extends Portfolio

ContentPortfolio defines a collection of content elements.

Method Index

• CBL_getPrefCModel()

Get the preferred content model.

Methods

• CBL_getPrefCModel

```
public abstract CModel CBL_getPrefCModel()
```

Get the preferred content model.

Interface com.adobe.cbl.Portfolio

public interface **Portfolio**

Portfolio defines a collection of content or layout elements.

Each Portfolio object represents a particular collection of application content or layout, such as a single FrameMaker file.

All objects in a Portfolio are represented as **PFElements**.

A **PFElement** is retrieved with a **Site** and a *semantic name*. If a Portfolio contains an element that is associated with the Site, it is required to return that element when `CBL_getElementBySite` is called. The Portfolio is also required to try and provide the element with the semantic name specified, or the closest approximation (up the Model tree), but if it can't, it can return the element with its preferred semantic name. See **PFElement** for details.

A group site allows collections of sites to cross-cut the normal content or layout hierarchy. If `CBL_checkSite()` returns `SiteUsage.GROUP`, it means that, for this Portfolio, this Site does not reference a **PFElement** but rather an unordered collection of other Sites.

The constructor for the object that implements this interface should take a **Referer** as a parameter. The Portfolio will use the **Referer** interface to create Sites. The default implementation of the **BindingSpec** class implements **Referer**.

Site, **PFElement**, **Referer**, **BindingSpec**

Method Index

- **CBL_addSiteToGroup(Site, Site)**

Add a site to a group site.

Groups are not allowed to contain self-references at any level.

- **CBL_checkSite(Site)**

Check if Site is valid, return its usage.

- **CBL_close()**

Close the portfolio.

Can be used for underlying file management, DB session unlocking, etc.

- **CBL_commit()**

Commit the portfolio.

Can be used to commit a transaction, do a Save of the file, etc.

- (String)
Delete an attribute from this portfolio.
- **CBL_enumElementsByNickname(String)**
List elements by an informal description.
- **CBL_enumGroups()**
Get all group sites.

Groups may have multiple binding sites, but this method only returns one site per group.
- **CBL_enumModelRestrictions()**
Get all model bindings imposed on this Portfolio instance.
- **CBL_enumRIDs()**
List all known Referer identifiers.
- **CBL_enumSites()**
List all the binding sites in this portfolio.
- **CBL_enumSitesInGroup(Site)**
Get all sites in a group.
- **CBL_getContainerElement(Site)**
Get the element that contains sub-content associated with Site.
- **CBL_getElementBySite(Site, String)**
Get the element associated with a Site.
- **CBL_getPortfolioAttr(String)**
Get an attribute on this portfolio.
- **CBL_getProp(String)**
Get a property.
- **CBL_getReferer()**
Get Referer for this instance.

Referer is an interface implemented by some object that holds references to this portfolio.
- **CBL_getRoot()**
Get root PORTFOLIO element.

All portfolios may contain references to other portfolios with the Model.PORTFOLIO element, which may be nothing more than a holder for top-level elements.
- **CBL_imposeSiteOnGroup(Site, Site)**
Impose a Site on an existing group.

Use this method when you want to impose an existing Site (e.g., from a BindingSpec) on a group.

- **CBL_listProps()**
Listproperties.
- **CBL_negotiateSemanticName(PFElement, Vector)**
Negotiatenewsemanticnameforelement.
- **CBL_newGroup()**
Createanemptynewgroupofsites.
- **CBL_newGroup(SiteEnumeration)**
Createanewgroupofsites.
- **CBL_open()**
Opentheportfolio.

Canbeusedforunderlyingfilemanagement,DBsessionandlocking,etc.
- **CBL_enumAttrs()**
Enumeratealloftheattributenamesinthisportfolio.
- **CBL_restrictModel(Site,String)**
Setamodelbinding.

APortfolioinstancecanrestrictthemodel(defaultsemanticname)foragivenSite.
- **CBL_setPortfolioAttr(String,Object)**
Setanattributeonthisportfolio,withdefaultretention(true).

Sameas

`CBL_setPortfolioAttr(attr, val, true)`
- **CBL_setPortfolioAttr(String,Object,boolean)**
Setanattributeonthisportfoliowithretainflag.
- **CBL_setProp(String,String)**
Setaproperty.

Methods

• CBL_open

```
public abstract void CBL_open() throws OpenFailed_CBLException
```

Opentheportfolio.

Canbeusedforunderlyingfilemanagement,DBsessionandlocking,etc.Anyattempttousemethodsdefinedinthis

interface, other than the Property Management methods, before a successful open has occurred, should raise a Runtime error.

Throws: OpenFailed_CBLEException
Portfolioopenfailed.

● CBL_commit

```
public abstract void CBL_commit() throws CommitFailed_CBLEException
```

Committheportfolio.

Canbeusedtocommitatransaction,doaSaveofthefile,etc.AnynewSiteassociationsaremadepermanent.

Throws: CommitFailed_CBLEException
Portfoliocommitfailed.

● CBL_close

Closetheportfolio.

Canbeusedforunderlyingfilemanagement,DBsessionunlocking,etc.AnynewSiteassociationsaremade permanent.

● CBL_getProp

```
public abstract String CBL_getProp(String prop) throws  
UnsupportedFeature_CBLEException
```

Getaproperty.

Parameters:

prop-The name of the property. The length of the String is required to be less than the value of the "propname.maxlength" property.

Returns:

The value of the property. Returns null if there is no such property.

Throws: UnsupportedFeature_CBLEException
This property is not supported.

● CBL_setProp

```
public abstract void CBL_setProp(String prop,  
String val) throws UnsupportedFeature_CBLEException
```

Setaproperty.

Parameters:

prop-Thenametheproperty.ThelengthoftheStringisrequiredtobelessthanthevalueofthe "propname.maxlength"property.
 val-The newvalueforthe property,which maybenull.

Throws: UnsupportedOperationException

Thispropertycannotbesetbyclients,orunrecognized.

● **CBL_listProps**

```
public abstract Enumeration CBL_listProps()
```

Listproperties.

Returns:

Anenumerationofpropertynames.Enumerationwithnoelementsifpropertylistisempty,nullifpropertylists arenotsupported.

● **CBL_setPortfolioAttr**

```
public abstract void CBL_setPortfolioAttr(String attr,
                                           Object val,
                                           boolean retain) throws
```

UnsupportedFeature_CBLException

Setanattributeonthisportfoliowithretainflag.

Parameters:

attr-Thenametheattribute.ThelengthoftheStringisrequiredtobelessthanthevalueofthe "attrname.maxlength"property.
 val-Thevalue,whichmaybenull.
 retain-Trueifattributeshouldberetainedfromonesessiontothenext(see CBL_open()).Falseifattributeis temporary(deletedon CBL_close()).

Throws: UnsupportedOperationException

Someimplementationsmaynotsupportattributesonportfolios.

● **CBL_setPortfolioAttr**

```
public abstract void CBL_setPortfolioAttr(String attr,
                                           Object val) throws
```

UnsupportedFeature_CBLException

Setanattributeonthisportfolio,withdefaultretention(true).

Sameas

```
CBL_setPortfolioAttr(attr, val, true)
```


Parameters:

attr-The name of the attribute. The length of the String is required to be less than the value of the "attrname.maxlength" property.
val-The value.

Throws: `UnsupportedFeature_CBLEException`

Some implementations may not support attributes on portfolios.

● CBL_getPortfolioAttr

`UnsupportedFeature_CBLEException`

Get an attribute on this portfolio.

Parameters:

attr-The name of the attribute. The length of the String is required to be less than the value of the "attrname.maxlength" property.

Returns:

The value of the attribute. Returns null if there is no such attribute on the Portfolio.

Throws: `UnsupportedFeature_CBLEException`

Some implementations may not support attributes on portfolios.

● CBL_portfolioAttributes

```
public abstract Enumeration CBL_portfolioAttributes()
```

Enumerate all of the attribute names in this portfolio.

Parameters:

retained-True lists attributes that are retained, false lists attributes that aren't.

Returns:

The elements of the enumeration are String objects. Enumeration with no elements if there are no attributes, null if attributes are not supported.

● CBL_deletePortfolioAttr

```
public abstract void CBL_deletePortfolioAttr(String attr) throws  
UnsupportedFeature_CBLEException
```

Delete an attribute from this portfolio.

Parameters:

attr-The name of the attribute. The length of the String is required to be less than the value of the "attrname.maxlength" property. The attribute is required to be on this Portfolio instance.

Throws: `UnsupportedFeature_CBLEException`

Some implementations may not support attributes on portfolios.

● CBL_enumRIDs

```
public abstract Enumeration CBL_enumRIDs()
```

List all known Referer identifiers.

Returns:

An enumeration of String, each is an RID that is in use in one or more Sites in this portfolio, in an undefined order. Can return null.

● CBL_enumSites

```
public abstract SiteEnumeration CBL_enumSites()
```

List all the binding sites in this portfolio.

Returns:

All binding sites in an undefined order. Can return null.

● CBL_checkSite

```
public abstract SiteUsage CBL_checkSite( Site aSite)
```

Check if Site is valid, return its usage.

Parameters:

aSite - The Site to check.

Returns:

Type of usage (type enumeration). Returns null if the Site is not used in this Portfolio.

● CBL_getElementBySite

```
public abstract PFElement CBL_getElementBySite( Site aSite,  
                                                String aName)
```

Get the element associated with a Site.

Parameters:

aSite - Required to be associated with an element in this Portfolio, i.e., CBL_checkSite() returns SiteUsage.ELEMENT.
aName - The desired semantic name, null to set the default name to the preferred name.

Returns:

The element returned will either be conformed to aName, or the default name will be the closest compatible ancestor model of aName, or the default name will be the preferred semantic name.

● CBL_getContainerElement

```
public abstract PFElementCBL_getContainerElement( SitesubSite)
```

Gettheelementthatcontainssub-contentassociatedwithSite.

Parameters:

subSite-Requiredto be associated with sub-content in this Portfolio, i.e., CBL_checkSite() returns
SiteUsage.DATA.

Returns:

Theelementthatcontainsthesub-contentassociatedwithsubSite.

● **CBL_enumElementsByNickname**

```
public abstract Enumeration CBL_enumElementsByNickname(String nickname)
```

Listelementsbyaninformaldescription.

Parameters:

nickname-Theinformaldescriptiondefinedbyauser.TheStringlengthisrequiredtobelessthanthevalueof
the"nickname.maxLength"property.

Returns:

AnenumerationofPFElementswhoseinformaldescriptionsareequal(case-sensitivestringequality)to
nickname.Returnsnullifnoelementshavematchinginformaldescriptions.

● **CBL_newGroup**

```
public abstract SiteCBL_newGroup( SiteEnumerationsites)
```

Createanewgroupofsites.

Parameters:

sites-Alistofsitestoformintoagroup.AllsitesarerequiredtobevalidforthisPortfolio.

Returns:

Anewsitethatcontainsthesites.

● **CBL_newGroup**

```
public abstract SiteCBL_newGroup()
```

Createanemptynewgroupofsites.

Returns:

Anewsitethatisempty(nomembersingroup).

● **CBL_imposeSiteOnGroup**

```
public abstract void CBL_imposeSiteOnGroup(Site group,
                                           SiteaSite)
```

Impose a Site on an existing group.

Use this method when you want to impose an existing Site (e.g., from a BindingSpec) on a group. The Portfolio must never have seen this site before and must internalize it (must not generate the same Site for a new usage).

Parameters:

group-Required to be a group site, i.e., CBL_checkSite() returns SiteUsage.GROUP.
aSite-Use the specified Site. Required to be a Site that has never been used before in this Portfolio, i.e., Portfolio.CBL_checkSite() returns null.

● **CBL_enumSitesInGroup**

```
public abstract SiteEnumeration CBL_enumSitesInGroup( Sitegroup)
```

Get all Sites in a group.

Parameters:

group-A Site that may be a group.

Returns:

An enumeration of all the sites in the group. This does a shallow enumeration only. If you want to enumerate nested groups, you need to apply this method again to those Sites that return SiteUsage.GROUP from CBL_checkSite(). Returns null if the Site specified by the "group" parameter is not a group.

● **CBL_enumGroups**

```
public abstract SiteEnumeration CBL_enumGroups()
```

Get all group sites.

Groups may have multiple binding sites, but this method only returns one site per group. The sites selected for each group is up to the implementation.

Returns:

An enumeration of all the sites that are groups of other sites in this portfolio. Returns null if there are no groups in this Portfolio.

● **CBL_addSiteToGroup**

```
Sitegroup,  
SiteaSite)
```

Add a site to a group site.

Groups are not allowed to contain self-references at any level. The implementation will throw an IllegalArgumentException exception if a cyclical reference is found after a deep enumeration of all Sites, including nested groups.

Parameters:

group-Required to be a group site.

aSite- The site to add to the group. The Site must not introduce a cyclical reference (required). This method does nothing if a Site is already in group.

● **CBL_getRoot**

```
public abstract PFElement CBL_getRoot()
```

Get root PORTFOLIO element.

All portfolios may contain references to other portfolios with the Model.PORTFOLIOelement, which may be nothing more than a holder for top-level elements. One PORTFOLIOelement is distinguished as the root element of a portfolio tree. Root elements never have data (PFElement.CBL_hasData() always returns false). Root elements cannot have negotiated (default) semantic names different from their preferred name of "Model.PORTFOLIO". This method returns the distinguished root PORTFOLIOelement.

Returns:

The root PORTFOLIOelement.

● **CBL_enumModelRestrictions**

Get all model bindings imposed on this Portfolio instance.

Returns:

An enumeration of ModelBindings. Returns null if there are no ModelBindings in this Portfolio.

● **CBL_restrictModel**

```
public abstract void CBL_restrictModel(Site aSite,  
                                       String aName)
```

Set a model binding.

A Portfolio instance can restrict the model (default semantic name) for a given Site. This is an optimization only and the Portfolio is not required to preserve this information during a CBL_commit().

Parameters:

aSite- The Site to restrict. Required to be a valid site for this Portfolio.

aName- The semantic name to impose.

● **CBL_getReferer**

```
public abstract Referer CBL_getReferer()
```

Get Referer for this instance.

Referer is an interface implemented by some object that holds references to this portfolio. Sites are created in the

context of a Referer.

● CBL_negotiateSemanticName

```
public abstract PFElement CBL_negotiateSemanticName( PFElement orig,  
                                                    Vector names)
```

Negotiate new semantic name for element.

To get an element with a desired content/layout model, use this method to negotiate for an element by referencing an instance of the element. The model is contained within the semantic names. Imposed model restrictions are ignored.

Parameters:

orig- The original element. Required to be an element in this Portfolio.

names- The desired semantic names, in order of preference. Required to have at least one element, and all elements must be String objects.

Returns:

A new instance of the element conformed to one of the specified names (set as the default semantic name for the element). Returns null if unable to conform to any of the specified semantic names, nor to any ancestor model or any of the semantic names.

Class com.adobe.cbl.Corner

```
java.lang.Object
|
+----com.adobe.cbl.Corner
```

public class **Corner**

extends Object

Corner defines an enumeration of rectangular orientations.

From template described in [<www.javaworld.com/javaworld/jw-07-1997/jw-07-enumerated.html>](http://www.javaworld.com/javaworld/jw-07-1997/jw-07-enumerated.html) . This enumeration supports naming and runtime enumeration.

Field Index

- **BOTTOM_LEFT**

xincreasestotheright,yincreasesup.

- **BOTTOM_RIGHT**

xincreasestothelleft,yincreasesup.

- **TOP_LEFT**

xincreasestotheright,yincreasesdown.

- **TOP_RIGHT**

xincreasestothelleft,yincreasesdown.

Constructor Index

- `Corner(String)`

Method Index

- `getEnum()`

GetanEnumeration.

- `toString()`

Returnthedeclaredidentifierstring.

Fields

- **TOP_LEFT**

```
public static final CornerTOP_LEFT
```

xincreasestotheright,yincreasesdown.

- **TOP_RIGHT**

```
public static final CornerTOP_RIGHT
```

xincreasestothelleft,yincreasesdown.

- **BOTTOM_LEFT**

```
public static final CornerBOTTOM_LEFT
```

xincreasestotheright,yincreasesup.

- **BOTTOM_RIGHT**

```
public static final CornerBOTTOM_RIGHT
```


x increases to the left, y increases up.

Constructors

● Corner

```
public Corner(String anId)
```

Methods

● getEnum

```
public static Enumeration getEnum()
```

Get an Enumeration.

● toString

```
public String toString()
```

Return the declared identifier string.

Overrides:

toString in class Object

Class com.adobe.cbl.Env

```
java.lang.Object
|
+----java.util.Dictionary
      |
      +----java.util.Hashtable
            |
            +----com.adobe.cbl.Env
```

public class **Env**

extends Hashtable

Constructor Index

• **Env()**

Construct an empty environment.

Constructors

• **Env**

public **Env()**

Construct an empty environment.

Class com.adobe.cbl.DimensionPro

```
java.lang.Object
|
+----com.adobe.cbl.DimensionPro
```

public class **DimensionPro**

extends Object

DimensionPro is a dimension for publishing professionals.

See Also:

Measure

Field Index

- height
- width

Constructor Index

- **DimensionPro**(double,double,Measure.Unit)
Construct a dimension.

Method Index

• `getUnit()`

Get the unit of measure.

Fields

• `height`

`public double height`

• `width`

`public double width`

Constructors

• `DimensionPro`

```
public DimensionPro(double h,  
                    double w,  
                    Measure. Unit aUnit)
```

Construct a dimension.

Parameters:

`h`-Height. Required to be nonnegative.

`w`-Width. Required to be nonnegative.

`aUnit`-Measuring unit.

Methods

• `getUnit`

public Measure. Unit getUnit()

Gettheunitofmeasure.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

DirectBinding

DirectBinding defines direct bindings.

Every direct binding has a collection of source sites and a collection of destination sites. The models for these sites are either content or layout. The interpretation of how the sites are to be combined, and

(refer to See Also section below).

Every direct binding effects only those elements associated with the destination sites AND any elements that are children of those elements: peers that follow the destination sites are explicitly not affected. For example, in a FlowTxtCModel content, a section (S1) that contains three paragraphs (P1 thru P3), can

the user wants P2 and P3 to have a different style, he must have direct bindings for both P2 and P3 -- applying an override to P2 only would have no effect on P3, which means P3 would still have defPS style inherited from the direct binding to the section.

Bindings are usually mandatory, meaning that all source and destination sites must be available and must be ignored, or the entire binding is ignored if no valid pair of sites is available.

See Also:

PlacementBinding, StyleBinding

Method Index

- **CBL_checkSitePFTType(Site)**

Check binding for Site and Portfolio type.

Used to discover if a particular Site is used in this binding, and what type of Portfolio it is expected to be in.

- **CBL_getDstSites()**

Get destination sites.

The interpretation of these destination sites depends on the class implementing this interface: e.g., for `PlacementBindings` the sites are the layout placement directives for the corresponding contents

- **CBL_getRequired()**

Check if binding is required.

- **CBL_getSrcSites()**

Get source sites.

The interpretation of these source sites depends on the class implementing this interface: e.g., for `PlacementBindings` the sites are the contents to be placed in the layout somewhere.

- **(boolean)**

Set binding requirement.

Methods

- **CBL_getSrcSites**

```
public abstract SiteEnumeration CBL_getSrcSites()
```

Get source sites.

The interpretation of these source sites depends on the class implementing this interface: e.g., for `PlacementBindings` the sites are the contents to be placed in the layout somewhere.

Returns:

The elements of the Enumeration are `Site` objects. This is a copy of the contents sites: edits made to the list of sites after this method returns are not reflected in the original enumeration.

- **CBL_getDstSites**

```
public abstract SiteEnumeration CBL_getDstSites()
```

Get destination sites.

The interpretation of these destination sites depends on the class implementing this interface: e.g., for `PlacementBindings` the sites are the layout placement directives for the corresponding contents.

Returns:

The elements of the Enumeration are Site objects. This is a copy of the layout sites: edits made to the list of sites after this method returns are not reflected in the original enumeration.

● CBL_checkSitePFType

```
public abstract PortfolioType CBL_checkSitePFType( Site aSite)
```

Check binding for Site and Portfolio type.

Used to discover if a particular Site is used in this binding, and what type of Portfolio it is expected to be in.

Parameters:

aSite - The binding site to check for.

Returns:

The type of Portfolio the Site is expected to be in, null if a Site is not in this binding.

● CBL_getRequired

```
public abstract boolean CBL_getRequired()
```

Check if binding is required. Determines what should be done if any source or destination binding is not present in their respective Portfolios. If the binding is required (true, the default value), it is an error for the Portfolio to not contain the binding. If the binding is optional (this method returns false), the binding site can be ignored as if it were not present in the binding. If there are no valid source or no valid destination bindings, the entire binding can be ignored.

Returns:

True (default) if binding is required, false if it is optional.

● CBL_setRequired

```
public abstract void CBL_setRequired(boolean required)
```

Set binding requirement.

Parameters:

required - True if binding is required, false if it is optional.

SeeAlso:

CBL_getRequired

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Interface com.adobe.cbl.DocGfxCModel

DocGfxCModel

extends GfxCModel

DocGfxCModel defines final form document graphics content.

See Also:

CModel, GfxCModel

Field Index

• PAGE

Identifies a page.

Fields

● PAGE

Identifies a page.

Interface com.adobe.cbl.GfxCModel

GfxCModel

extends CModel

GfxCModel defines graphic content.

See Also:

CModel

Field Index

• FIGURE

Identifies graphic contents.

• TEXTURE

Identifies a texture.

Fields

● FIGURE

```
public static final String FIGURE
```

Identifies graphic contents.

● TEXTURE

```
public static final String TEXTURE
```

Identifies a texture.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Class com.adobe.cbl.DocSpec

```
java.lang.Object
|
+----com.adobe.cbl.DocSpec
```

public class **DocSpec**

extends Object

DocSpec defines a dynamic document.

Constructor Index

- **DocSpec()**
Create an empty DocSpec
- **DocSpec(ContentPortfolio, LayoutPortfolio, BindingSpec)**
Create a complete DocSpec.

Method Index

- **getBindingSpec()**
Get the binding specification.
- **getContentPortfolio()**
Get the content portfolio.
- **getLayoutPortfolio()**

Get the layout portfolio.

- **setBindingSpec**(BindingSpec)

Set the binding specification portfolio.

- **setContentPortfolio**(ContentPortfolio)

Set the content portfolio.

- **setLayoutPortfolio**(LayoutPortfolio)

Set the layout portfolio.

Constructors

- **DocSpec**

```
public DocSpec (ContentPortfolio cp,  
                LayoutPortfolio lp,  
                BindingSpec bs)
```

Create a complete DocSpec.

Parameters:

cp - Can be null.

lp - Can be null.

bs - Can be null.

- **DocSpec**

```
public DocSpec ()
```

Create an empty DocSpec

Methods

- **getContentPortfolio**

```
public ContentPortfolio getContentPortfolio ()
```

Get the content portfolio.

Returns:

Can be null, which means this DocSpec is incomplete.

● **getLayoutPortfolio**

```
public LayoutPortfolio getLayoutPortfolio()
```

Get the layout portfolio.

Returns:

Can be null, which means this DocSpec is incomplete.

● **getBindingSpec**

```
public BindingSpec getBindingSpec()
```

Get the binding specification.

Returns:

Can be null, which means this DocSpec is incomplete.

● **setContentPortfolio**

```
public void setContentPortfolio(ContentPortfolio cp)
```

Set the content portfolio.

Parameters:

cp - Can be null.

● **setLayoutPortfolio**

```
LayoutPortfolio lp)
```

Set the layout portfolio.

Parameters:

lp - Can be null.

● setBindingSpec

```
public void setBindingSpec (BindingSpecbs)
```

Setthebindingspecificationportfolio.

Parameters:

bs-Canbenull.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Interface

com.adobe.cbl.DynStylesLModel

DynStylesLModel

extends `StylesLModel`

DynStylesLModel defines dynamic media presentation information.

See Also:

`LModel`

Field Index

• DIRECTION

Identifies a playback direction.

•

Identifies a playback rate.

Fields

• RATE

Identifies a playback rate.

• DIRECTION

```
public static final String DIRECTION
```

Identifies a playback direction.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Interface

com.adobe.cbl.DynamicsLModel

DynamicsLModel

extends LModel

DynamicsLModel defines timing information.

See Also:

LModel

Field Index

• END

Identifies end time.

• EVENT

Identifies an event.

• START

Identifies start time.

• TIMER

Identifies a timer.

Fields

• TIMER

```
public static final String TIMER
```

Identifiesatimer.

● EVENT

public static final String EVENT

Identifiesanevent.

● START

public static final String START

Identifiesastarttime.

● END

public static final String END

Identifiesandendtime.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Interface com.adobe.cbl.LModel

LModel

extends [Model](#)

LModel is the base class at the root of a hierarchy of layout models.

Each subclass defines a layout model by virtue of its position in the tree. For example:

`LModel -> PlacementLModel`

A *PlacementLModel* can be treated as a *LModel* by virtue of its inheritance from that interface. In this settle on a particular type.

Field Index

• LAYOUT

Identifies all sub-interfaces as layout models

Fields

• LAYOUT

`public static final String LAYOUT`

Identifies all sub-interfaces as layout models

Class com.adobe.cbl.ElementBinding

```
java.lang.Object
|
+----com.adobe.cbl.ElementBinding
```

public class **ElementBinding**

extends Object

ElementBinding implements `globalelementbindings`.

Element bindings have a semantic name (which self-describes its content model) and one or more `FlowTxtCModel.PARA` elements, by default.

Constructor Index

- **ElementBinding(String, Vector)**
Construct an element binding.

Method Index

- **equals(Object)**
Override the `equals()` method for special processing.
- **getDfltSemanticName()**
Get the semantic name.
- **getDfltSites()**
Get the binding sites for the layout elements associated with this element.

Constructors

● ElementBinding

```
public ElementBinding(String name,  
                      Vector sites)
```

Construct an element binding.

Parameters:

name- The semantic name to be given a default binding. Required to be a CModel.
sites- All elements are required to be Site objects from the same Layout Portfolio. The contents of the Vector are copied.

Methods

● getDfltSemanticName

```
public String getDfltSemanticName()
```

Get the semantic name.

Returns:

Semantic name with Model prefix. The semantic name is required to be a CModel.

● getDfltSites

```
public SiteEnumeration getDfltSites()
```

Get the binding sites for the layout elements associated with this element.

Returns:

The elements of the enumeration are Site objects. Required to be present in the Layout Portfolio associated with the BindingSpec from which this ElementBinding was obtained.

● equals

```
public boolean equals(Object obj)
```

Override the equals() method for special processing. Two Element Bindings are equal if their default semantic names are equal.

Overrides:

equals in class Object

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

FlowTxtCModel

extends TxtCModel

FlowTxtCModel defines structured text that can flow.

Any element that directly contains content may have subsites: binding sites which apply to only *part* of the raw content, such as italicized word "part" above for the element that would contain this entire paragraph.

See Also:

[TxtCModel](#)

Field Index

- **ARTICLE**

A sequence of sections, with an optional heading (title).

- **FOOTNOTE**

A footnote.

- **HEADING**

A heading for an article, section, or list.

- **LABEL**

A text label.

- **LIST**

A list of items, with an optional heading.

- **LIST_ITEM**

An item in a list.

• PARA

A unit of text.

• SECTION

A logical division, with an optional heading.

Fields

● ARTICLE

```
public static final String ARTICLE
```

A sequence of sections, with an optional heading (title).

● SECTION

```
public static final String SECTION
```

A logical division, with an optional heading.

● LIST

A list of items, with an optional heading.

● LIST_ITEM

```
public static final String LIST_ITEM
```

An item in a list.

● PARA

A unit of text.

● HEADING

```
public static final String HEADING
```

Aheadingforanarticle,section,orlist.

● LABEL

```
public static final String LABEL
```

Atextlabel.

● FOOTNOTE

```
public static final String FOOTNOTE
```

Afootnote.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Interface com.adobe.cbl.TxtCModel

TxCModel

extends CModel

TxCModel defines text content.

See Also:

CModel

Field Index

• TCODE

Identifies text encoded data, such as a JavaScript

• TEXT

Identifies any kind of human readable text content

Fields

• TEXT

Identifies any kind of human readable text content

• TCODE

```
public static final String TCODE
```

Identifies text encoded data, such as a JavaScript

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Interface

com.adobe.cbl.FontStylesLModel

FontStylesLModel

extends `StylesLModel`

FontStylesLModel defines font presentation information.

See Also:

`LModel`

Field Index

• CHARMAP

Identifies a character mapping

• FONT

Identifies a font

Fields

• FONT

Identifies a font

• CHARMAP

```
public static final String CHARMAP
```

Identifies a character mapping

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Interface

com.adobe.cbl.GfxStylesLModel

GfxStylesLModel

extends StylesLModel

GfxStylesLModel defines graphics presentation information.

See Also:

LModel

Field Index

• FILL

Identifies a fill style.

• LINE

Identifies a line style.

Fields

• LINE

Identifies a line style.

• FILL

Identifiesafillstyle.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Class com.adobe.cbl.IntervalPro

```
java.lang.Object
|
+----com.adobe.cbl.IntervalPro
```

public class **IntervalPro**

extends Object

IntervalPro is a timing interval for multimedia professionals.

Should be able to handle timecode format, as well as absolute and relative times based on some measure.
struct?) and units (int).

See Also:

Measure

Constructor Index

- **IntervalPro**(String,String,Measure.Unit)
Construct an interval.

Method Index

- **getUnit**()
Get the unit of measure.

Constructors

IntervalPro

```
public IntervalPro(String startTime,  
                  String endTime,  
                  Measure. Unit aUnit)
```

Construct an interval.

Methods

getUnit

```
public Measure. Unit getUnit()
```

Get the unit of measure.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Interface

LayeredGfxCModel

extends `GfxCModel`

LayeredGfxCModel defines layered graphics content.

See Also:

`CModel`, `GfxCModel`

Field Index

• **LAYER**

Identifies a layer.

Fields

● **LAYER**

```
public static final String LAYER
```

Identifies a layer.

LayoutPortfolio

extends Portfolio

LayoutPortfolio defines a collection of layout elements.

Method Index

- **CBL_getPrefLModel()**

Get the preferred layout model

Methods

- **CBL_getPrefLModel**

```
public abstract LModel CBL_getPrefLModel()
```

Get the preferred layout model

Interface com.adobe.cbl.LinkCModel

LinkCModel

extends CModel

LinkCModel defines hyperlink content.

See Also:

CModel

Field Index

• LINK

Identifies a link

Fields

● LINK

Identifies a link

Interface com.adobe.cbl.Measure

Measure

Measure defines units of measurement.

Java 1.1 enables this funky sort of "enum" equivalent, by using a static inner class in the interface! This technique was first explored by Dan Craft.

Field Index

- - Centimeters.
- **INCH**
 - Inches.
- **PIXEL**
 - Picture elements.
- **POINT**
 - Printer's point: 72.27pi.
- **PSPOINT**
 - PostScript point: 72pi
- **SAMPLE**
 - Digital sample; audio, etc.

Fields

● INCH

```
public static final Measure. UnitINCH  
    Inches.
```

● POINT

```
public static final Measure. UnitPOINT  
    Printer'spoint:72.27pi.
```

● PSPOINT

```
public static final Measure. UnitPSPOINT  
    PostScriptpoint:72pi
```

● CENTIMETER

```
public static final Measure. UnitCENTIMETER  
    Centimeters.
```

● PIXEL

```
public static final Measure. UnitPIXEL  
    Pictureelements.
```

● SAMPLE

```
public static final Measure. UnitSAMPLE  
    Digitalsample;audio,etc.
```

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Class com.adobe.cbl.ModelBinding

```
java.lang.Object
|
+----com.adobe.cbl.ModelBinding
```

public class **ModelBinding**

extends Object

ModelBinding restricts a Site to a particular semantic name.

Allows a document author to restrict the model of a particular Site. Records the intention that the author wishes elements of that Site to always be of a specified model.

See Also:

[BindingSpec](#)

Constructor Index

• **ModelBinding**(String, Site)

Construct a restricted model binding.

Method Index

• **getRestrictedSemanticName**()

Get the semantic name that restricts the Site.

The type of Portfolio that the Site is expected to be found in is implied by the model of the semantic name, i.e., ContentPortfolio for CModel and LayoutPortfolio for LModel.

• `getRestrictedSite()`

Get the Site to be restricted.

Constructors

• `ModelBinding`

```
public ModelBinding(String name,  
                    Site site)
```

Construct a restricted model binding.

Parameters:

site - The Site to be restricted.

name - The semantic name that restricts the Site.

Methods

• `getRestrictedSite`

```
public Site getRestrictedSite()
```

Get the Site to be restricted.

• `getRestrictedSemanticName`

```
public String getRestrictedSemanticName()
```

Get the semantic name that restricts the Site.

The type of Portfolio that the Site is expected to be found in is implied by the model of the semantic name, i.e., Content Portfolio for CModel and Layout Portfolio for LModel.

Class com.adobe.cbl.ModelHelper

```
java.lang.Object
|
+----com.adobe.cbl.ModelHelper
```

public class **ModelHelper**

extends Object

defines utility routines for Model interfaces.

This code sucks and needs to be made more efficient.

See Also:

Model

Constructor Index

• **ModelHelper()**

Method Index

• **checkConformance(Class, PFElement, int)**

Test an element against the namespace of a model.

• **checkName(Class, String)**

Test a full semantic name against the namespace of a model.

• **getNameSpace(Class)**

Returns an array of the semantic names implemented by this model.

Test an element against the namespace of a model.

Parameters:

aModel- The model to test against.

anElement- The element to test.

pref-PFElement.PREF if the test should be performed on the element's preferred semantic name, PFElement.DFLT if the test should be performed on the element's default semantic name.

Returns:

True if the element is defined in this namespace. Inheritance of namespaces is taken into account.

🔍 **checkName**

```
public static boolean checkName(Class aModel,  
                                String aName)
```

Test a full semantic name against the namespace of a model.

Parameters:

aModel- The model to test against.

aName- The full semantic name to test.

Returns:

True if the full semantic name is defined in this namespace. Inheritance of namespaces is taken into account.

🔍 **parseModel**

```
public static String parseModel(String aName)
```

Extract the model from the full semantic name.

Parameters:

aName- The full semantic name to parse. Required to be a well-formed semantic name.

Returns:

The model part of the full name.

🔍 **parseElement**

`public static String parseElement(String aName)`

Extracttheelementnamefromthefullsemanticname.

Parameters:

aName-Thefullsemanticnametoparse.Requiredtobeawell-formedsemanticname.

Returns:

Theelementpartofthefullname.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Interface com.adobe.cbl.MultiGfxCModel

MultiGfxCModel

extends GfxCModel

defines multiple-mode graphics content.

See Also:

CModel, GfxCModel

Field Index

•

Identifies an artwork view.

• PREVIEW

Identifies a preview.

• WIREFRAME

Identifies a wireframe view.

Fields

● PREVIEW

```
public static final String PREVIEW
```

Identifies a preview.

● ARTWORK

public static final String ARTWORK

Identifiesanartworkview.

● WIREFRAME

public static final String WIREFRAME

Identifiesawireframeview.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Interface com.adobe.cbl.PFElement

PFElement

PFElement is a Portfolio element.

All objects in a Portfolio are represented as **PFElements**. These semantics of an element is defined in the context of a content or layout model; a *semantic name* is a label for these semantics. You can think of the semantic name as the type of the element, within the domain of content and layout models -- this type is element, e.g., Paragraph vs. Heading.

The current theory is that Sites can be created from scratch (in the context of a Portfolio), or added to an element, or examined on an element, but cannot be removed from an element. This is to help prevent "dangling references" that might occur from BindingSpecs that still refer to the deleted Site.

Fundamentally, the model for a PFElement object is that it is a reference to the underlying content or layout data, which is identified uniquely by a Site. The underlying data has an intrinsic intended usage, which is called it's "preferred" semantic name. This preferred name is reflected in the PFElement object.

The default semantic name is set by the Portfolio when the PFElement is created. It is the semantic name `CBL_getElementBySite()`, or if this element is a child element that was retrieved implicitly through enumeration of the children of an ancestor element that was name. The PFElement remembers whether it provided the default semantic name as requested or not (see `CBL_isAsIntended()`).

The attribute methods in PFElement refer to attributes on the underlying data, not in the ephemeral PFElement object.

Two PFElement objects (e1 and e2) from the same Portfolio, with the same Site, same preferred semantic name, and same reference, are equal, i.e. (e1.equals(e2)) is true. However, the expression (e1 == e2) may be false. The actual implementation of this object must conform to these equality requirements.

Summarizing the data requirements for a PFElement:

Name	Description	Reflects underlying data	Affectsequals()
------	-------------	--------------------------	-----------------

Site	Refereridand uniquereference id	X	X
Defaultsemantic name	Oneof:requested semanticname, "inherited" semanticname, preferred.		X
Preferred semanticname	Modeland elementname.	X	X
Reference	Tounderlying data.	X	X
Attributes	Get&Set	X	
Portfolio	Thatthiselement comesfrom.	X	X
Children	Ofthiselement.	X	X

See Also:

Model, Site, Portfolio

Field Index

- **DFLT**

Defaultsemanticname.

-

Attributenamefornickname.

- **PREF**

Preferredsemanticname.

Method Index

- **CBL_deleteElementAttr(String)**

Deleteanattributefromthiselement.

- **CBL_elementAttributes()**

Enumerate all of the attribute names on this element.

- **enumSubSites()**
List all the Sites associated with this element.
- **CBL_enumSubSites()**
List all the data (sub-content) Sites associated with this element.
- **CBL_estimateDataSize(Site)**
Estimate the size of the data.

The size returned is a conservative count.
- **CBL_getBytes(String, Site)**
Get element data as an array of bytes.
- **CBL_getChild(PFElement)**
Get a child in sequence.

Get the child after the specified child.
- **CBL_getDataFormat(Site)**
Get the preferred format for unstructured data.
- **CBL_getElementAttr(String)**
Get an arbitrary attribute on the element.
- **CBL_getInputStream(String, Site)**
Get element data as an InputStream (bytes).

Valid for all models.
- **CBL_getInts(String, Site)**
Get element data as an array of ints (32-bit).

Valid for all models.
- **CBL_getNickname()**
Get an informal description.

IMPLEMENTATION NOTE: If your implementation supports the general attribute methods listed above, you are encouraged to implement informal descriptions with an attribute named

"description.informal" (you can use the **NICKNAME** constant defined above.)

- **CBL_getObject(Class,Site)**

GetelementdataasanObject.

ClassaClassdependsontheelement'smodel:

GfxCModel

RectPro,returnsthegraphicssub-rectangle.

StyleLModel

Anapplicationspecificstyleobject.

- **CBL_getObjectType(Site)**

Getthepreferredobjecttypeofstructureddata.

- **CBL_getOrigSite()**

GettheSiteusedtoretrievethiselement.

- **CBL_getParent()**

Getparent.

- **CBL_getPortfolio()**

GetthePortfoliothatthiselementcamefrom.

- **CBL_getReader(String,Site)**

GetelementdataasaReader(Characters).

ValidforTxtCModelelementsonly.

IMPLEMENTATIONNOTE: AllPFElementimplementationsarerequiredtoimplementthis methodforTxtCModels.

- **CBL_getSemanticName(int)**

Getthesemanticnamedusedtoretrievethiselement.

- **CBL_getString(String,Site)**

GetelementdataasaString.

ValidforTxtCModelandLinkCModelelementsonly.

- **CBL_getSubRange(Site)**

Get range used to define sub-content.

Valid for TxtCModel elements only.

- **CBL_getSubStyleNames()**

List the supported sub-style names.

- **CBL_hasChildren()**

Any children?

- **CBL_hasData()**

Check if data can be retrieved from this element.

- **CBL_imposeSite(Site)**

Impose a Site on this element.

Use this method when you want to impose an existing Site (e.g., from a BindingSpec) on an element.

- **CBL_isAsIntended()**

Test this element to see if it has the intended semantic name.

Elements are normally acquired by doing a lookup in a Portfolio with a Site and a semantic name.

- **CBL_isStructuredData()**

Check if the data is structured or unstructured.

- **CBL_newSite()**

Create a new binding site on this element from scratch.

The Portfolio will generate a new UID for the Site, and use the RID that the Portfolio was created with.

- **(Object, RectPro, Site)**

Create/impose binding site on graphics sub-content.

Use this only on elements whose preferred semantic name comes from the
its subinterfaces.

GfxCModel or one of

- **CBL_newSubStyleSite(Object, String, Site)**

Create/impose binding site on a sub-style.

Use this only on elements whose preferred semantic name comes from the
its subinterfaces.

StylesLModel or one of

- **CBL_newSubTxtSite**(Object,int,int,Site)

Create/imposebindingsiteontextsub-content.

Use this only on elements whose preferred semantic name comes from the
its subinterfaces.

TxtCModel or one of

- **CBL_newSubTxtSite**(Object,int,Site)

Create/imposebindingsitetoendof textsub-content.

Use this only on elements whose preferred semantic name comes from the
its subinterfaces.

TxtCModel or one of

- **CBL_setElementAttr**(String,Object,boolean)

Set arbitrary attribute on the element.

- **CBL_setNickname**(String)

Set informal description.

IMPLEMENTATION NOTE: If your implementation supports the general attribute methods listed above, you are encouraged to implement informal descriptions with an attribute named "description.informal" (you can use the **NICKNAME** constant defined above.)

Fields

- **DFLT**

```
public static final int DFLT
```

Default semantic name.

- **PREF**

```
public static final int PREF
```

Preferred semantic name.

- **NICKNAME**

```
public static final String NICKNAME
```

Attributenamefornickname.

Methods

● CBL_setElementAttr

```
public abstract void CBL_setElementAttr(String attr,  
                                         Object val,  
                                         boolean retain) throws
```

UnsupportedFeature_CBLException

Setarbitraryattributeontheelement.

Parameters:

attr-Theattributename.TheStringlengthisrequiredtobelessthanthevalueofthe
Portfolioproperty"attrname.maxlength".
val-Theattributevalue.Canbenull.
retain-Trueifattributeshouldberetainedfromonesessiontothenext(see
Portfolio.CBL_open()).Falseifattributeistemporary(deletedon Portfolio.CBL_close()).

Throws: UnsupportedFeature_CBLException

Someimplementationsmaynotsupportattributesonelements.

● CBL_getElementAttr

```
public abstract Object CBL_getElementAttr(String attr) throws  
UnsupportedFeature_CBLException
```

Getarbitraryattributeontheelement.

Parameters:

attr-Theattributename.TheStringlengthisrequiredtobelessthanthevalueofthe
Portfolioproperty"attrname.maxlength".

Returns:

Theattributevalue.Returnsnullifthereisnosuchattributeontheelement.

Throws: UnsupportedFeature_CBLException

Someimplementationsmaynotsupportattributesonelements.

● CBL_elementAttributes

```
public abstract Enumeration CBL_elementAttributes() throws  
UnsupportedFeature_CBLException
```

Enumeratealloftheattributenamesonthiselement.

Returns:

TheelementsoftheenumerationareStringobjects.Returnsnulliftherearenoattributeson
thiselement.

Throws: UnsupportedFeature_CBLException

Someimplementationsmaynotsupportattributesonelements.

● CBL_deleteElementAttr

```
public abstract void CBL_deleteElementAttr(String attr) throws  
UnsupportedFeature_CBLException
```

Deleteanattributefromthiselement.

Parameters:

attr-The nameoftheattribute.The lengthoftheStringisrequiredtobelessthanthevalue
ofthe"attrname.maxLength"property.TheattributeisrequiredtobeonthisPFElement
instance.

Throws: UnsupportedFeature_CBLException

Someimplementationsmaynotsupportattributesonelements.

● CBL_getNickname

```
public abstract String CBL_getNickname()
```

Getinformaldescription.

IMPLEMENTATIONNOTE: Ifyourimplementationsupportsthegeneralattributemethods
listedabove,youareencouragedtoimplementinformaldescriptionswithanattributenamed
"description.informal"(youcanusethe **NICKNAME**constantdefinedabove.)

Returns:

Aninformaldescriptiondefinedbyauser.Thesystemmakesnoguaranteesaboutthe

consistency or semantics of these descriptions. Returns null if there is no nickname on this element.

● CBL_setNickname

```
public abstract void CBL_setNickname(String nickname)
```

Set informal description.

IMPLEMENTATION NOTE: If your implementation supports the general attribute methods listed above, you are encouraged to implement informal descriptions with an attribute named "description.informal" (you can use the **NICKNAME** constant defined above.)

Parameters:

nickname- The informal description to associate. Can be null, which deletes any existing nickname on this element. The length of the nickname String is required to be less than the value of the Portfolio property "nickname.maxlength".

● CBL_newSite

```
public abstract Site CBL_newSite()
```

Create a new binding site on this element from scratch.

The Portfolio will generate a new UID for the Site, and use the RID that the Portfolio was created with.

● CBL_imposeSite

```
public abstract void CBL_imposeSite(Site aSite)
```

Impose a Site on this element.

Use this method when you want to impose an existing Site (e.g., from a BindingSpec) on an element. The Portfolio from which this element was derived must not have seen this site before, and must internalize (must not generate the same Site for a new usage).

Parameters:

aSite- Use the specified Site. Required to be a Site that has never been used before in the Portfolio from which this element was derived, i.e., Portfolio.CBL_checkSite() returns null.

● CBL_getOrigSite

```
public abstract SiteCBL_getOrigSite()
```

Get the Site used to retrieve this element.

Returns:

Returns null if this element was retrieved indirectly, i.e., as the child of some ancestor element.

● **CBL_enumBindingSites**

```
public abstract SiteEnumerationCBL_enumBindingSites()
```

List all the Sites associated with this element.

Returns:

An enumeration of Sites, excluding sub-sites, associated directly with this element (does not include any groups which may eventually refer to this element). Returns null if there are no binding sites on this element.

● **CBL_enumSubSites**

```
public abstract SiteEnumerationCBL_enumSubSites()
```

List all the data (sub-content) Sites associated with this element.

Returns:

An enumeration of sub-sites associated with the data for this element (does not include any groups which may eventually refer to this element). Returns null if there are no binding sites in the data of this element.

● **CBL_newSubTxtSite**

```
public abstract SiteCBL_newSubTxtSite(Object origData,  
                                         int beginIndex,  
                                         int endIndex,  
                                         Site aSite)
```

Create/impose binding site context sub-content.

Use this only on elements whose preferred semantic name comes from the `TxtCModel` or one of

its subinterfaces. Check the semantic name before proceeding.

Parameters:

origData-The originalelementcontentobject.
beginIndex-Index of first character. Required to be nonnegative and less than endIndex.
endIndex-Index of last character, plus one. Required to be nonnegative.
aSite-Site to impose on this element, or null to get a new Site. If non-null, required to be a Site that has never been used before in the Portfolio instance from which this element was derived.

Returns:

Same as aSite if aSite was non-null, otherwise it is the new Site.

● **CBL_newSubTxtSite**

```
public abstract Site CBL_newSubTxtSite(Object origData,  
                                       int beginIndex,  
                                       Site aSite)
```

Create/impose bindings site to end of text sub-content.

Use this only on elements whose preferred semantic name comes from the TxtCModel or one of its subinterfaces. Check the semantic name before proceeding.

Parameters:

origData-The originalelementcontentobject.
beginIndex-Index of first character. Required to be nonnegative.
aSite-Site to impose on this element, or null to get a new Site. If non-null, required to be a Site that has never been used before in the Portfolio instance from which this element was derived.

Returns:

Same as aSite if aSite was non-null, otherwise it is the new Site.

● **CBL_newSubGfxSite**

```
public abstract Site CBL_newSubGfxSite(Object origData,  
                                       RectProaRect,  
                                       Site aSite) throws
```

UnsupportedFeature_CBLException

Create/impose binding site on graphic sub-content.

Use this only on elements whose preferred semantic name comes from the
its subinterfaces. Check the semantic name before proceeding.

GfxCModel or one of

The requested rectangle should be clipped to the largest enclosing boundary of the source graphic.

Parameters:

origData-The original element content object.
aRect-A sub-rectangle of the graphic. Required to be well-formed.
aSite-Site to impose on this element, or null to generate a new Site. If non-null, required to be a Site that has never been used before in the Portfolio instance from which this element was derived.

Returns:

Same as a Site if a Site was non-null, otherwise it is the new Site.

Throws: UnsupportedOperationException

Cannot bind to sub-rectangles of graphics.

● **CBL_newSubStyleSite**

```
public abstract Site CBL_newSubStyleSite(Object origData,  
                                           String field,  
                                           Site aSite) throws
```

UnsupportedOperationException

Create/impose bindings site on a sub-style.

Use this only on elements whose preferred semantic name comes from the
its subinterfaces. Check the semantic name before proceeding.

StylesLModel or one of

See CBL_getSubStyleNames().

Parameters:

origData-The original element content object.
field-Name of the sub-style field.
aSite-Site to impose on this element, or null to generate a new Site. If non-null, required to be a Site that has never been used before in the Portfolio instance from which this element was derived.

Returns:

Same as a Site if a Site was non-null, otherwise it is the new Site.

Throws: UnsupportedOperationException

This specified field name does not refer to a supported sub-style.

● CBL_getSubStyleNames

List the supported sub-style names. Use this only on elements whose preferred semantic name comes from the StylesLMModel or one of its subinterfaces. Check the semantic name before proceeding.

Returns:

The Hashtable keys are the field names for sub-styles, and the values are String objects that contain a UI description of the field. Returns null if this feature is not supported.

● CBL_hasChildren

```
public abstract boolean CBL_hasChildren()
```

Any children?

● CBL_getChild

```
public abstract PFElement CBL_getChild( PFElement after)
```

Get a child in sequence.

Get the child after the specified child.

Parameters:

after-Specify null to get the first child. If non-null, required to be a PFElement that was returned by a call to this.CBL_getChild().

● CBL_getParent

```
public abstract PFElement CBL_getParent()
```

Get parent.

Returns:

Parent element. If null, element is the Portfolio root.

● CBL_getPortfolio

```
public abstract Portfolio CBL_getPortfolio()
```

Get the Portfolio that this element came from.

● CBL_getSemanticName

```
public abstract String CBL_getSemanticName(int which)
```

Get the semantic name used to retrieve this element.

Parameters:

which-PREF returns the preferred semantic name. DFLT returns the default semantic name.

● CBL_isAsIntended

```
public abstract boolean CBL_isAsIntended()
```

Test this element to see if it has the intended semantic name.

Elements are normally acquired by doing a lookup in a Portfolio with a Site and a semantic name. The Portfolio is not obligated to return an element with the specified semantic name--only an element with the specified Site. If the returned semantic name matches the suggested, this method returns true, otherwise it returns false.

Returns:

True if element's semantic name matches the original semantic names suggested in the original retrieval.

● CBL_hasData

```
public abstract boolean CBL_hasData()
```

Check if data can be retrieved from this element.

Returns:

True if data can be retrieved from this element, false if data cannot be retrieved from this element.

● CBL_isStructuredData

```
public abstract boolean CBL_isStructuredData()
```

Check if the data is structured or unstructured.

Returns:

True if data is structured, false if data is unstructured. This element must have data (CBL_hasData() is true) (required). Even if this method returns true, there are some models, such as DynCModel, for which the unstructured methods may still be used. This means you can either get the data in its internal runtime encoding (as a `StructureObject`), or in its external file stream encoding (unstructured).

● CBL_estimateDataSize

```
public abstract int CBL_estimateDataSize(SiteSubSite)
```

Estimate the size of the data.

The size returned is a conservative count. Actual size may be somewhat less than size returned, but never greater.

Parameters:

subSite-Reference to sub-content, null for all content. If non-null, required to be a Site that is valid for the Portfolio instance from which this element was derived.

Returns:

Size in bytes. A return value of 0 means that there is no data, or that the data is structured, or that it would be too costly to count the data (and the clients should use one of the stream data retrieval methods).

● CBL_getDataFormat

```
public abstract String CBL_getDataFormat(SiteSubSite)
```

Get the preferred format for unstructured data.

Parameters:

subSite-Reference to sub-content, null for all content. If non-null, required to be a Site that is valid for the Portfolio instance from which this element was derived.

Returns:

The MIME type of the data. Returns null if data is not unstructured or if the element has no data.

● CBL_getObjectType

```
public abstract Class CBL_getObjectType(SitesubSite)
```

Get the preferred object type of structured data.

Parameters:

subSite-Reference to sub-content, null for all content. If non-null, required to be a Site that is valid for the Portfolio instance from which this element was derived.

Returns:

The Class of the structured object. Returns null if data is not structured or if element has no data.

● CBL_getObject

```
public abstract Object CBL_getObject(Class aClass,  
                                     SitesubSite)
```

Get element data as an Object.

Class aClass depends on the element's model:

GfxCModel

RectPro, return the graphic sub-rectangle.

StyleLModel

An application specific style object.

Parameters:

aClass-The class the caller expects the returned Object to be. Pass null to indicate that the preferred class is acceptable.

subSite-Reference to sub-content, null for all content. If non-null, required to be a Site that is valid for the Portfolio instance from which this element was derived.

Returns:

Object Object of type aClass, or null if this can't return an object of a Class type.

● CBL_getString

```
public abstract String CBL_getString(String mimeType,
                                     SitesubSite)
```

GetelementdataasaString.

ValidforTxtCModelandLinkCModelelementsonly.

Parameters:

mimeType-Theexpecteddataencoding/format.Passnulltoindicate that the preferred format is acceptable.

subSite-Referencetosub-content,nullforallcontent.Ifnon-null,requiredtobe a Site that is valid for the Portfolio instance from which this element was derived.

Returns:

Theelementdata,ornullifthiscan'treturnthedataasaString,and/orthiscan'treturnthe data in the specified format.

● CBL_getInputStream

```
public abstract InputStream CBL_getInputStream(String mimeType,
                                               SitesubSite)
```

GetelementdataasanInputStream(bytes).

Validforallmodels.Interpretationofreturnvaluedependsontheelement'smodel:

GfxCModel

Alwaysacomplete/standalonegraphicdatastream.IfsubSiteisnon-null,thegraphicis croppedtothesub-rectanglespecifiedwhenthesubSitewasassociated.Otherwise,itisthe completegraphic.

TxtCModel

Alwaysastreamoftextdata.IfsubSiteisnull,thestreamreturnsallofthetextdata, otherwiseitjustreturnstheportionassociatedwithsubSite.

DynCModel

Alwaysacomplete/standaloneclipdatastream.IfsubSiteisnon-null,theclipistrimmedto thebeginandendtimespecifiedwhenthesubSitewasassociated.Otherwise,itisthe completeclip.

IMPLEMENTATIONNOTE: AllPFEelementimplementationsarerequiredtoimplementthis methodforallmodels.Wewantclientstorelyonthismethodworkingforallunstructureddata,

assuming the mimeType is acceptable.

Parameters:

mimeType-The expected data encoding/format. Pass null to indicate that the preferred format is acceptable.

subSite-Reference to sub-content, null for all content. If non-null, required to be a Site that is valid for the Portfolio instance from which this element was derived.

Returns:

The element data, or null if this can't return the data as an InputStream, and/or this can't return the data in the specified format.

● **CBL_getReader**

```
public abstract Reader CBL_getReader(String mimeType,  
                                     Site subSite)
```

Get element data as a Reader (Characters).

Valid for TxtCModel element only.

IMPLEMENTATION NOTE: All PFElement implementations are required to implement this method for TxtCModels.

Parameters:

mimeType-The expected data encoding/format. Pass null to indicate that the preferred format is acceptable.

subSite-Reference to sub-content, null for all content. If non-null, required to be a Site that is valid for the Portfolio instance from which this element was derived.

Returns:

The element data, or null if this can't return the data as a Reader, and/or this can't return the data in the specified format.

● **CBL_getBytes**

```
public abstract byte[] CBL_getBytes(String mimeType,  
                                    Site subSite)
```

Get element data as an array of bytes. Valid for all models. Interpretation of return value depends on the element's model:

GfxCModel

Always a complete/standalone graphic byte array. If subSite is non-null, the graphic is cropped to the sub-rectangle specified when the subSite was associated. Otherwise, it is the complete graphic.

TxtCModel

Always a byte array of text data. If subSite is null, the byte array contains all of the text data, otherwise it just contains the portion associated with subSite.

DynCModel

Always a complete/standalone clip byte array. If subSite is non-null, the clip is trimmed to the begin and end times specified when the subSite was associated. Otherwise, it is the complete clip.

Parameters:

contentType-The expected data encoding/format. Pass null to indicate that the preferred format is acceptable.

subSite-Reference to sub-content, null for all content. If non-null, required to be a Site that is valid for the Portfolio instance from which this element was derived.

Returns:

The element data, or null if this can't return the data as a byte array, and/or this can't return the data in the specified format.

● **CBL_getInts**

```
public abstract int[] CBL_getInts(String contentType,  
                                Site subSite)
```

Get element data as an array of ints (32-bit).

Valid for all models. Interpretation of return value depends on the element's model:

GfxCModel

Always a complete/standalone graphic word array. If subSite is non-null, the graphic is cropped to the sub-rectangle specified when the subSite was associated. Otherwise, it is the complete graphic.

TxtCModel

Always a word array of text data, where only the low-order 16 bits of the word are used (upper 16 are set to zero). If subSite is null, the word array contains all of the text data, otherwise it just contains the portion associated with subSite.

DynCModel

Always a complete/standalone clip word array. If subSite is non-null, the clip is trimmed to the begin and end times specified when the subSite was associated. Otherwise, it is the complete clip.

Parameters:

contentType-The expected data encoding/format. Pass null to indicate that the preferred format is acceptable.

subSite-Reference to sub-content, null for all content. If non-null, required to be a Site that is valid for the Portfolio instance from which this element was derived.

Returns:

The element data, or null if this can't return the data as an int array, and/or this can't return the data in the specified format.

● **CBL_getSubRange**

```
public abstract int[] CBL_getSubRange(Site subSite)
```

Get range used to define sub-content.

Valid for TxtCModel elements only. Use this to find the location of sub-content relative to the rest of the content in the node.

Parameters:

subSite-Required to be a valid reference to sub-content in this element.

Returns:

The beginIndex is <ret>[0] and endIndex is <ret>[1].

Class com.adobe.cbl.PFElementWrapper

java.lang.Object

|

public class **PFElementWrapper**

extends Object

implements PFElement

PFElementWrapper is a wrapper class for PFElement objects.

The CBL API requires that PFElement objects returned through various methods (such as Portfolio.CBL_getElementBySite())

multiple binding sites, see PFElement.CBL_getOrigSite()), or then negotiated default semantic name (see Portfolio.CBL_negotiateSemanticName()). On the other hand, the structured Portfolio data is expected to be invariant across all references for parent/child relationships, attributes on elements, data contents, etc.

In order to simplify implementation of Portfolio and PFElement classes, this wrapper class is provided. It allows an implementation to create a single database tree of elements for a Portfolio, but return PFElement objects as needed with instance specific data. These PFElementWrapper objects keep instance data relevant only their particular creation, and also to the original object.

See Also:

PFElement

Field Index

• orig

The true, original element in the Portfolio.

Constructor Index

• PFElementWrapper(PFElement, Site)

Construct with preferred name.

• PFElementWrapper(PFElement, Site, String, boolean)

Construct from scratch.

Method Index

- `getSite()` (String)
- `CBL_elementAttributes()`
- `CBL_enumBindingSites()`
- `CBL_enumSubSites()`
- `CBL_estimateDataSize(Site)`
- `CBL_getBytes(String,Site)`
- `CBL_getChild(PFElement)`
- `CBL_getDataFormat(Site)`
- `CBL_getElementAttr(String)`
- `CBL_getInputStream(String,Site)`
- `CBL_getInts(String,Site)`
- `CBL_getNickname()`
- `CBL_getObject(Class,Site)`
- `CBL_getObjectType(Site)`
- `CBL_getOrigSite()`
Get the Site used to retrieve this element.

This method uses instanced data and does not reference the original element.
- `CBL_getParent()`
- `CBL_getPortfolio()`
- `getSite()` (String,Site)
- `CBL_getSemanticName(int)`
Get the semantic name used to retrieve this element.

This method uses instanced data and does not reference the original element.
- `CBL_getString(String,Site)`
- `CBL_getSubRange(Site)`
- `CBL_getSubStyleNames()`

- **CBL_hasChildren()**

- **CBL_hasData()**

- **CBL_imposeSite(Site)**

- **CBL_isAsIntended()**

Test this element to see if it has the intended semantic name.

Elements are normally acquired by doing a lookup in a Portfolio with a Site and a semantic name.

- **CBL_isStructuredData()**

- **CBL_newSite()**

CBL_newSite

- **CBL_newSubGfxSite(Object, RectPro, Site)**

- **CBL_newSubStyleSite(Object, String, Site)**

- (Object, int, int, Site)

- (Object, int, Site)

- **CBL_setElementAttr(String, Object, boolean)**

- **CBL_setNickname(String)**

- **dummy()**

Need a method here for the section comment.

- **isValidSite(PFElement, Site)**

Utility to check if a binding site is present on an element.

Fields

- **orig**

public final PFElement orig

The true, original element in the Portfolio.

Constructors

- **PFElementWrapper**

public PFElementWrapper(PFElement orig,
Site origSite,

String name,
boolean asIntended)

Construct from scratch.

Parameters:

orig- The true, original element.
origSite- The original site used to acquire this element. Can be null. Required to be a binding site on orig if non-null.
name- The default semantic name.
asIntended- The name is the requested name.

● **PFElementWrapper**

```
public PFElementWrapper(PFElement orig,  
                          Site origSite)
```

Construct with preferred name.

Parameters:

orig- The true, original element.
origSite- The original site used to acquire this element.

Methods

● **isValidSite**

```
PFElement anElem,  
Site aSite)
```

Utility to check if a binding site is present on an element.

Parameters:

anElem- The PFElement to test.
aSite- The binding site to test.

● **dummy**

```
public void dummy()
```

Need a method here for the section comment.

● **CBL_setElementAttr**

```
public void CBL_setElementAttr(String attr,  
                                Object val,  
                                boolean retain) throws
```

● CBL_getElementAttr

```
public Object CBL_getElementAttr(String attr) throws UnsupportedFeature_CBLErrorException
```

● CBL_elementAttributes

```
public Enumeration CBL_elementAttributes() throws UnsupportedFeature_CBLErrorException
```

● CBL_deleteElementAttr

```
public void CBL_deleteElementAttr(String attr) throws
```

● CBL_getNickname

```
public String CBL_getNickname()
```

● CBL_setNickname

● CBL_newSite

```
public SiteCBL_newSite()
```

CBL_newSite

● CBL_imposeSite

SiteaSite)

● CBL_getOrigSite

```
public SiteCBL_getOrigSite()
```

GettheSiteusedtoretrievethiselement.

Thismethodusesinstancedataanddoesnotreferencetheoriginalelement.

Returns:

Returnsnullifthiselementwasretrievedindirectly,i.e.,asthechildofsomeancestorelement.

● CBL_enumBindingSites

```
public SiteEnumerationCBL_enumBindingSites()
```

● CBL_enumSubSites

```
public SiteEnumerationCBL_enumSubSites()
```

● CBL_newSubTxtSite

```
public SiteCBL_newSubTxtSite(ObjectorigData,  
                             int beginIndex,  
                             int endIndex,  
                             SiteaSite)
```

● CBL_newSubTxtSite

```
public SiteCBL_newSubTxtSite(ObjectorigData,  
                             int beginIndex,  
                             SiteaSite)
```

● CBL_newSubGfxSite

```
public SiteCBL_newSubGfxSite(ObjectorigData,  
                             RectProaRect,  
                             SiteaSite)throws      UnsupportedFeature_CBLErrorException
```

● CBL_newSubStyleSite

```
public SiteCBL_newSubStyleSite(ObjectorigData,  
                                String field,  
                                SiteaSite)throws      UnsupportedFeature_CBLErrorException
```

● CBL_getSubStyleNames

```
public Hashtable CBL_getSubStyleNames()
```

● CBL_hasChildren

```
public boolean CBL_hasChildren()
```

● CBL_getChild

```
public PFElementCBL_getChild( PFElementafter)
```

● CBL_getParent

```
public PFElementCBL_getParent()
```

● CBL_getPortfolio

```
public Portfolio CBL_getPortfolio()
```

● CBL_getSemanticName

Getthesemanticnamedusedtoretrievethiselement.

Thismethodusesinstancedataanddoesnotreferencetheoriginaelement.

Parameters:

which-PREFreturnsthepreferredsemanticname.DFLTreturnsthedefaultsemanticname.

● CBL_isAsIntended

```
public boolean CBL_isAsIntended()
```

Testthiselementtoseeifithastheintendedsemanticname.

ElementsarenormallyacquiredbydoingalookupinaPortfoliowithaSiteandasemanticname.ThePortfolioisnot obligatedtoreturnanelementwiththespecifiedsemanticname--onlyanelementwiththespecifiedSite.Ifthe returnedsemanticnamematchesthesuggested,thismethodreturnstrue,otherwiseitreturnsfalse.

Thismethodusesinstancedataanddoesnotreferencetheoriginaelement.

Returns:

Trueifelement'ssemanticnamematchestheoriginalsemanticnamesuggestedintheoriginalretrieval.

● CBL_hasData

```
public boolean CBL_hasData()
```

● CBL_isStructuredData

● CBL_estimateDataSize

```
public int CBL_estimateDataSize(SitesubSite)
```

● CBL_getDataFormat

```
public String CBL_getDataFormat(SitesubSite)
```

● CBL_getObjectType

```
public Class CBL_getObjectType(Site subSite)
```

● CBL_getObject

```
SitesubSite)
```

● CBL_getString

```
SitesubSite)
```

● CBL_getInputStream

```
public InputStream CBL_getInputStream(String mimeType,  
                                       SitesubSite)
```

● CBL_getReader

```
SitesubSite)
```

● CBL_getBytes

```
public byte[] CBL_getBytes(String mimeType,  
                           SitesubSite)
```

● CBL_getInts

```
SitesubSite)
```

● CBL_getSubRange

```
public int[] CBL_getSubRange(SitesubSite)
```

```
java.lang.Object
|
+----com.adobe.cbl.PFEnumeration
```

public class **PFEnumeration**

extends Object

implements Enumeration

PFEnumeration is a Portfolio element enumeration utility class.

Enumerates all children of a Portfolio depth first, with parents reported before children, e.g., if a Portfolio has a root element R with two children A and B, and A has child AA, and B and two children B1 and B2, the enumeration would produce: R, A, AA, B, B1, B2.

Constructor Index

- **PFEnumeration(Portfolio)**

Method Index

- **hasMoreElements()**
More elements?
- **nextElement()**
Get the next element.
- **resetEnumeration()**

Reset enumeration.

Once the enumeration is exhausted, this object will continue to throw `NoSuchElementException` until it is reset.

Constructors

● `PFEnumeration`

```
public PFEnumeration(Portfolio pf)
```

Methods

● `hasMoreElements`

```
public boolean hasMoreElements()
```

More elements?

● `nextElement`

```
public Object nextElement() throws NoSuchElementException
```

Get the next element.

Returns:

The next element.

Throws: `NoSuchElementException`

No more elements left.

● `resetEnumeration`

```
public void resetEnumeration()
```

Reset enumeration.

Once the enumeration is exhausted, this object will continue to throw `NoSuchElementException`

until it is reset.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Interface

com.adobe.cbl.PlacedTxtCModel

PlacedTxtCModel

extends `TxtCModel`

PlacedTxtCModel defines `placedtextcontent`.

See Also:

`TxtCModel`

Field Index

• ARTTEXT

Identifies text start, such as text on a curve.

• PRE

Preformatted text.

Fields

• ARTTEXT

```
public static final String ARTTEXT
```

Identifies text start, such as text on a curve.

• PRE

```
public static final String PRE
```

Preformattedtext.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Class `com.adobe.cbl.PlacementBinding`

```
java.lang.Object
|
+----com.adobe.cbl.PlacementBinding
```

public class **PlacementBinding**

extends `Object`

implements `DirectBinding`

implements placement information for content elements.

Placement bindings aggregate one to many content elements with one to many layout elements. The

element. If the current content element does not completely fill the layout element, and there are more content elements, the placement continues with the next content element. Repeat until all layout elements are exhausted.

Constructor Index

• **PlacementBinding**(`SiteEnumeration`,`SiteEnumeration`)

Construct a required placement binding.

• **PlacementBinding**(`SiteEnumeration`,`SiteEnumeration`,`boolean`)

Construct a placement binding, specifying requirement.

SiteEnumeration places,
boolean isRequired)

Construct a placement binding, specifying requirement.

Parameters:

contents-All elements are required to be Site objects from the same Content Portfolio.
places-All elements are required to be Site objects from the same Layout Portfolio.
isRequired-True if binding is required, false if binding is optional.

Methods

● CBL_getSrcSites

```
public synchronized SiteEnumeration CBL_getSrcSites()
```

Get content sites from a content portfolio.

These are the contents to be placed according to the layout sites defined in this binding. The content sites are returned in FIFO order.

Returns:

The elements of the Enumeration are Site objects. This is a copy of the content sites: edits made to the list of sites after this method returns are not reflected in the original enumeration.

● CBL_getDstSites

```
public synchronized SiteEnumeration CBL_getDstSites()
```

Get placement sites from a layout portfolio. These are the placement directives to be used to position the contents in this binding. The layout sites are returned in FIFO order.

Returns:

The elements of the Enumeration are Site objects. This is a copy of the placement sites: edits made to the list of sites after this method returns are not reflected in the original enumeration.

● CBL_checkSitePFType

```
public synchronized PortfolioType CBL_checkSitePFType(Site aSite)
```

Check binding for site and portfolio type.

Used to discover if a particular site is used in this binding, and what type of portfolio it is expected to be in.

Parameters:

aSite - The binding site to check for.

Returns:

The type of portfolio the site is expected to be in, null if a site is not in this binding.

● CBL_getRequired

```
public synchronized boolean CBL_getRequired()
```

Check if binding is required. Determines what should be done if any source or destination binding is not present in their respective portfolios. If the binding is required (true, the default value), it is an error for the portfolio to not contain the binding. If the binding is optional (this method returns false), the binding site can be ignored as if it were not present in the binding. If there are no valid source or no valid destination bindings, the entire binding can be ignored.

Returns:

True (default) if binding is required, false if it is optional.

● CBL_setRequired

```
public synchronized void CBL_setRequired(boolean required)
```

Set binding requirement.

Parameters:

required - True if binding is required, false if it is optional.

See Also:

CBL_getRequired

Interface

com.adobe.cbl.PlacementLModel

PlacementLModel

extends LModel

defines layout location information.

See Also:

LModel

Field Index

- **BREAK**

Identifies a (line) break.

- **CONTAINER**

Identifies a container.

- **EXPLICIT**

Identifies explicit placement.

- **FLOW**

Identifies a flow.

- **LOCATION**

Identifies a location.

- **TEMPLATE**

Identifies a template.

Fields

● LOCATION

```
public static final String LOCATION
```

Identifies a location.

● CONTAINER

```
public static final String CONTAINER
```

Identifies a container.

● FLOW

Identifies a flow.

● TEMPLATE

```
public static final String TEMPLATE
```

Identifies a template.

● EXPLICIT

```
public static final String EXPLICIT
```

Identifies explicit placement.

● BREAK

```
public static final String BREAK
```

Identifies a (line) break.



[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Class com.adobe.cbl.PortfolioType

```
java.lang.Object
|
+----com.adobe.cbl.PortfolioType
```

public class **PortfolioType**

extends Object

PortfolioType defines an enumeration of PortfolioTypes.

From template described in [<www.javaworld.com/javaworld/jw-07-1997/jw-07-enumerated.html>](http://www.javaworld.com/javaworld/jw-07-1997/jw-07-enumerated.html). This enumeration supports naming and runtime enumeration.

Field Index

• CONTENT

Contenttype.

• LAYOUT

Layouttype.

Constructor Index

• PortfolioType(String)

Method Index

• getEnum()

GetanEnumeration.

• toString()

Returnthedeclaredidentifierstring.

Fields

● CONTENT

```
public static final PortfolioTypeCONTENT
```

Contenttype.

● LAYOUT

```
public static final PortfolioTypeLAYOUT
```

Layouttype.

Constructors

● PortfolioType

```
public PortfolioType(String anId)
```

Methods

● getEnum

```
public static Enumeration getEnum()
```

Get an Enumeration.

● toString

```
public String toString()
```

Return the declared identifier string.

Overrides:

toString in class Object

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Interface

com.adobe.cbl.RasterGfxCModel

RasterGfxCModel

extends GfxCModel

RasterGfxCModel defines raster/image graphics content.

See Also:

CModel, GfxCModel

Field Index

• CHANNEL

Identifies a channel.

• RASTER

Identifies a raster graphic.

Fields

● RASTER

```
public static final String RASTER
```

Identifies a raster graphic.

● CHANNEL

```
public static final String CHANNEL
```

Identifiesachannel.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

```
java.lang.Object
|
+----com.adobe.cbl.RectPro
```

public class **RectPro**

extends Object

RectPro is a rectangle for publishing professionals.

See Also:

Measure

Field Index

- height
- width
- x
- y

Constructor Index

- **RectPro**(double,double,double,double,Measure.Unit,Corner,Corner)
Construct a rectangle.
The rectangle is required to be well-formed.

Method Index

- **getCorner()**

Get the location of the corner of the rectangle.

- **getOrigin()**

Get the location of the origin of the coordinate system.

- **getUnit()**

Get the unit of measure.

Fields

- **x**

```
public double x
```

- **y**

```
public double y
```

- **width**

```
public double width
```

- **height**

```
public double height
```

Constructors

- **RectPro**

```
public RectPro(double x,  
               double y,
```



```
double w,  
double h,  
Measure. Unit aUnit,  
Corner o,  
Corner c)
```

Construct a rectangle.

The rectangle is required to be well-formed.

Parameters:

x- Horizontal distance of Corner from Origin.
y- Vertical distance of Corner from Origin.
w- Width, required to be nonnegative.
h- Height, required to be nonnegative.
aUnit- Unit of measure for x, y, w, and h.
o- Location of origin in coordinate system. This is described as a corner of the container (such as a page) that defines the origin. Positive x and positive y extend away from the origin into the container.
c- Location of corner of rectangle, to which x and y are offsets from the origin.

Methods

● getUnit

```
public Measure. Unit getUnit()
```

Get the unit of measure.

● getOrigin

```
public Corner getOrigin()
```

Get the location of the origin of the coordinate system.

● getCorner

```
public Corner getCorner()
```

Get the location of the corner of the rectangle.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Class `com.adobe.cbl.Site`

```
java.lang.Object
|
+----com.adobe.cbl.Site
```

public class **Site**

extends `Object`

Site defines a binding site descriptor.

Referer identifier (RID) and a unique reference identifier (UID). The RID identifies a unique referer to references, e.g., a `BindingSpec` which makes reference to a `ContentPortfolio` and a `LayoutPortfolio`. The UID is a unique, invariant identifier of a target reference, e.g., the unique id of an element in a `Portfolio`.

The scope of binding sites is defined by the scope of the RID implementation. For example, if RID's are implemented using Microsoft GUID's, the scope of binding sites is universal. The RID can have any encoding or syntax, as long as it is representable as a Java String -- however, many applications will find it convenient if the RID has a trivial encoding as 7-bit ASCII (0x20 thru 0x7e, inclusive), and even better, if the encoding excluded 0x22 (double quote), 0x27 (apostrophe), 0x26 (ampersand), and 0x3c (less than).

The scope of UID's need only be unique within a `Portfolio`. The UID must be representable as a Java String, and must not include the `UID_FORBIDDEN_CHAR` defined below.

See Also:

`Referer`

Field Index

- **rid**

TheRefereridentifier.

- **uid**

Theuniquereferenceidentifier.

- **UID_FORBIDDEN**

Uniquereferenceidentifiersareforbiddenfromcontainingthis character, intheformofasinglecharacterString.

- **UID_FORBIDDEN_CHAR**

Uniquereferenceidentifiersareforbiddenfromcontainingthis character.

Constructor Index

- **Site(String)**

CreateaSitefromacompositeString.

Giventhefollowingsetup:

```
SiteaSite=newSite(rid,uid);  
SitebSite=newSite(aSite.toString());
```

Arrangeforthefollowingexpressiontobetrue:

```
aSite.equals(bSite)&&bSite.equals(aSite)
```

- **Site(String,String)**

CreateaSite.

Method Index

- **equals(Object)**
Test forequality.
- **hashCode()**
Supplyan efficienthashingfunction.
- **toString()**
SupplyaStringvalueforthisobject.

Fields

• **UID_FORBIDDEN**

```
public static final String UID_FORBIDDEN
```

Uniquereferenceidentifiersareforbiddenfromcontainingthis character,intheformofasinglecharacterString.

• **UID_FORBIDDEN_CHAR**

```
public static final char UID_FORBIDDEN_CHAR
```

Uniquereferenceidentifiersareforbiddenfromcontainingthis character.

• **rid**

```
public final String rid
```

TheRefereridentifier.

• **uid**

```
public final String uid
```

The unique reference identifier.

Constructors

● Site

```
public Site(String rid,  
            String uid)
```

CreateaSite.

Parameters:

rid-Refereridentifier.
uid-Uniquereferenceidentifier.

● Site

```
public Site(String composite)
```

CreateaSitefromacompositeString.

Giventhefollowingsetup:

```
SiteaSite=newSite(rid,uid);  
SitebSite=newSite(aSite.toString());
```

Arrangeforthefollowingexpressiontobetrue:

```
aSite.equals(bSite)&&bSite.equals(aSite)
```

Parameters:

composite-CompositeofRIDandUID.

Methods

● equals

```
public boolean equals(Object anObject)
```

Test for equality.

Overrides:

equalsinclassObject

● hashCode

public int hashCode()

Supplyanefficienthashingfunction.

Overrides:

hashCodeinclassObject

● toString

public String toString()

SupplyaStringvalueforthisobject.

Overrides:

toStringinclassObject

Interface com.adobe.cbl.SiteEnumeration

SiteEnumeration

extends Enumeration

defines a type checked version of Enumeration.

Calls to `nextElement()` and `nextSite()` can be intermixed: calls to `nextSite()` are equivalent to casted calls to `nextElement()`, e.g., `(Site)nextElement()`.

Method Index

- **getArray()**

Get the Site enumeration as an array.

Always returns the full list of Sites, even if `nextElement()` has already been called on this instance.

- **hasMoreElements()**

More elements?

- **nextElement()**

Get the next element.

- **nextSite()**

Get the next Site.

Methods

- **hasMoreElements**

```
public abstract boolean hasMoreElements()
```


More elements?

● **nextElement**

```
public abstract Object nextElement() throws NoSuchElementException
```

Getthenextelement.

Returns:

Thenextelement.

Throws: NoSuchElementException

Nomoreelementsleft.

● **nextSite**

```
public abstract Site nextSite() throws NoSuchElementException
```

GetthenextSite.

Returns:

ThenextSite.

Throws: NoSuchElementException

Nomoreelementsleft.

● **getArray**

```
public abstract Site[] getArray()
```

GettheSiteenumerationasanarray.

AlwaysreturnsthefulllistofSites,evenifnextElement()hasalreadybeencalledonthisinstance.

Returns:

ArrayofSite.Returnsnulliftherearenoelementsintheenumeration,oritcan'tbe renderedasanarray.

Class com.adobe.cbl.SiteUsage

```
java.lang.Object
|
+----com.adobe.cbl.SiteUsage
```

public class **SiteUsage**

extends Object

SiteUsage defines an enumeration of usages for Sites.

From template described in [<www.javaworld.com/javaworld/jw-07-1997/jw-07-enumerated.html>](http://www.javaworld.com/javaworld/jw-07-1997/jw-07-enumerated.html) . This enumeration supports naming and runtime enumeration.

Field Index

• DATA

Associated with sub-content.

• ELEMENT

Associated with an element.

• GROUP

Associated with a group.

Constructor Index

• **SiteUsage**(String)

Method Index

- **getEnum()**

GetanEnumeration.

- **toString()**

Returnthedeclaredidentifierstring.

Fields

- **ELEMENT**

```
public static final SiteUsageELEMENT
```

Associatedwithanelement.

- **DATA**

```
public static final SiteUsageDATA
```

Associatedwithsub-content.

- **GROUP**

```
public static final SiteUsageGROUP
```

Associatedwithagroup.

Constructors

- **SiteUsage**

```
public SiteUsage(String anId)
```

Methods

• **getEnum**

```
public static Enumeration getEnum()
```

GetanEnumeration.

• **toString**

```
public String toString()
```

Returnthedeclaredidentifierstring.

Overrides:

toStringinclassObject

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Class com.adobe.cbl.StyleBinding

```
java.lang.Object
|
+----com.adobe.cbl.StyleBinding
```

public class **StyleBinding**

extends Object

implements DirectBinding

StyleBinding implements presentation information.

layout elements. An example of the former is assigning ParaStyle13 and CharStyle20 to a paragraph PARA:9. An example of the latter is assigning BorderStyle55 to a layout box called FRAME:3.

When there is one destination site and one source site, the source site is applied to the destination site. Otherwise, a union of all of the source sites is formed, and this union is applied to each of the destination sites. Both aggregations (sources and destinations) are unordered. Destination sites must all come from the same Portfolio, and be of the same type (all content or all layout).

Constructor Index

• **StyleBinding**(SiteEnumeration, SiteEnumeration)

Construct a required style binding for content.

• **StyleBinding**(SiteEnumeration, SiteEnumeration, boolean)

Construct a required style binding with designated targets.

• **StyleBinding**(SiteEnumeration, SiteEnumeration, boolean, boolean)

Construct a style binding with designated targets, indicating requirement.

Method Index

■ CBL_checkSitePFType(Site)

CheckbindingforSiteandPortfoliotype.

Used to discover if a particular Site is used in this binding, and what type of Portfolio it is expected to be in.

• CBL_getDstSites()

Getdestinationsites.

These are the sites that are the targets of the style to be applied.

▪ CBL_getRequired()

Checkifbindingisrequired.

• CBL_getSrcSites()

Getstylesitesfromalayoutportfolio.

• (boolean)

Setbindingrequirement.

• checkTargets()

Getinterpretationoftargetsites.

Seedescriptionofreturnvalues.

Constructors

StyleBinding

```
public StyleBinding(SiteEnumerationcontents,
                   SiteEnumerationstyles)
```

Constructarequiredstylebindingforcontent.

Parameters:

contents-All elements are required to be Site objects from the same Content Portfolio.

styles - All elements are required to be Site objects from the same LayoutPortfolio.

● StyleBinding

```
public StyleBinding(SiteEnumeration targets,  
                   SiteEnumeration styles,  
                   boolean isContents)
```

Constructarequiredstylebindingwithdesignatedtargets.

Parameters:

targets-AllelementsarerequiredtobeSiteobjectsfromthesameContentPortfolio(if
isContentstrue)orthesameLayoutPortfolio(ifisContentsfalse).
styles-AllelementsarerequiredtobeSiteobjectsfromthesameLayoutPortfolio.
isContents-Trueiftargetsarecontents,falseiftheyarelaysouts.

● StyleBinding

```
public StyleBinding(SiteEnumeration targets,  
                   SiteEnumeration styles,  
                   boolean isContents,  
                   boolean isRequired)
```

Constructastylebindingwithdesignatedtargets,indicatingrequirement.

Parameters:

targets-AllelementsarerequiredtobeSiteobjectsfromthesameContentPortfolio.
styles-AllelementsarerequiredtobeSiteobjectsfromthesameLayoutPortfolio.
isContents-Trueiftargetsarecontents,falseiftheyarelaysouts.
isRequired-Trueifbindingisrequired,falseifoptional.

Methods

● CBL_getSrcSites

```
public synchronized SiteEnumerationCBL_getSrcSites()
```

Getstylesitesfromalayoutportfolio.Thesearethestyledirectivestobeappliedtothetargets
specifiedinthisbinding.

Returns:

The elements of the Enumeration are Site objects. This is a copy of the style sites: edits made to the list of sites after this method returns are not reflected in the original enumeration.

● CBL_getDstSites

```
public synchronized SiteEnumeration CBL_getDstSites()
```

Get destination sites.

These are the sites that are the targets of the styles to be applied.

Returns:

The elements of the Enumeration are Site objects. This is a copy of the destination sites: edits made to the list of sites after this method returns are not reflected in the original enumeration. Required to all be in the same Portfolio, and to all be the same type (content or layout).

● CBL_checkSitePFType

```
public synchronized PortfolioType CBL_checkSitePFType( Site aSite)
```

Check binding for site and Portfolio type.

Used to discover if a particular site is used in this binding, and what type of Portfolio it is expected to be in.

Parameters:

aSite - The binding site to check for.

Returns:

The type of Portfolio the site is expected to be in, null if a site is not in this binding.

● CBL_getRequired

```
public synchronized boolean CBL_getRequired()
```

Check if binding is required. Determines what should be done if any source or destination binding is not present in their respective Portfolios. If the binding is required (true, the default value), it is an error for the Portfolio to not contain the binding. If the binding is optional (this method returns false), the binding site can be ignored as if it were not present in the binding. If there are no valid

source or no valid destination bindings, the entire binding can be ignored.

Returns:

True(default)ifbindingisrequired,falseifitoptional.

● **CBL_setRequired**

```
public synchronized void CBL_setRequired(boolean required)
```

Setbindingrequirement.

Parameters:

required-Trueifbindingisrequired,falseifitoptional.

SeeAlso:

CBL_getRequired

● **checkTargets**

```
public PortfolioTypecheckTargets()
```

Getinterpretationoftargetsites.

Seedescriptionofreturnvalues.

Returns:

CONTENTifCBL_getDstSites()returnsSitesfromaContentPortfolio,LAYOUTif
CBL_getContentSites()returnsSitesfromaLayoutPortfolio.

```
java.lang.Object
|
+----com.adobe.cbl.Template
```

public class **Template**

extends Object

Template implements `templatespecification`.

Template specifications are used to associate specific designs of layout templates (page masters, column

PlacementLModel.CONTAINER). However, in some cases, the layout template is computed as part of the formatting or pagination process; furthermore, many different templates may be used in the course of a single placement binding (e.g., put FlowTxtCModel.ARTICLE into PlacementLModel.FLOW). Each CompSequence may optionally specify a list of Template objects to record the set of templates that

The usages are orthogonal and extensible. Omission of a usage specification indicates that the composer has no intention for this template other than to indicate that this particular template instance is a valid result of template resolution, and not some other. Indication of a usage declares the composers intension, and formatters should try to comply. For example, indicating that a particular template is TemplateUsage.FIRST means that the composer wants to distinguish the look of the first page (or column, etc.) from the others.

Each usage may only appear once in every Template (it is an error to say Template.LEFT twice for the same Template). However, for a particular CompSequence, several Templates may have the same usage. For example, if three of five templates are marked as TemplateUsage.FIRST, it means that the first page

To indicate that the formatting client should have complete control over template resolution, use only a single Template object per CompSequence, and omit all usage associations. The single template is a reference to an algorithm or a chunk of state that is used to resolve templates.

See Also:

TemplateUsage, CompSequence

Field Index

• template

Bindingsitefortemplateelement.

• usage

Optionallistofusages.

Constructor Index

• Template(Site,Enumeration)

Constructwithatemplateandusagelist.

Fields

• template

```
public final Site template
```

Bindingsitefortemplateelement.

• usage

```
public final Vector usage
```

Optionallistofusages.AllareTemplateUsageobjects.

Constructors

● Template

```
public Template(Sitetemplate,  
                Enumeration usage)
```

Constructwithatemplateandusagelist.

Parameters:

template-Bindingsitefortemplateelement.

usage-AllelementsoftheEnumerationarerequiredtobeTemplateUsageobjects.Canbe null.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

```
java.lang.Object
|
+----com.adobe.cbl.TemplateUsage
```

public class **TemplateUsage**

extends Object

TemplateUsage defines an enumeration of usages for Templates.

This enumeration may be extended by subclassing.

From template described in [<www.javaworld.com/javaworld/jw-07-1997/jw-07-enumerated.html>](http://www.javaworld.com/javaworld/jw-07-1997/jw-07-enumerated.html) . This enumeration supports naming and runtime enumeration.

Field Index

- **FIRST**

First(page)template.

- **LAST**

Last(page)template.

- **LEFT**

Left-handtemplate.

-

Right-handtemplate.

Constructor Index

- **TemplateUsage(String)**

Construct the next element of the enumeration.

Method Index

- **equals(Object)**

Equality is based on the identifier string.

- **getEnum()**

Get an Enumeration.

- **toString()**

Return the declared identifier string.

Fields

- **FIRST**

```
public static final TemplateUsage FIRST
```

First (page) template.

- **LAST**

```
public static final TemplateUsage LAST
```

Last (page) template.

- **RIGHT**

```
public static final TemplateUsage RIGHT
```

Right-hand template.

● LEFT

```
public static final TemplateUsageLEFT
```

Left-handtemplate.

Constructors

● TemplateUsage

```
public TemplateUsage(String anId)
```

Constructthenextelementoftheenumeration.

Methods

● getEnum

```
public static Enumeration getEnum()
```

GetanEnumeration.

● toString

```
public String toString()
```

Returnthedeclaredidentifierstring.

Overrides:

toStringinclassObject

● equals

```
public boolean equals(Object obj)
```

Equalityisbasedontheidentifierstring.

Overrides:

`equalsinclassObject`

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)


```
java.lang.Object
|
+----java.lang.Throwable
      |
      +----java.lang.Exception
            |
            +----com.adobe.cbl.CBLEException
```

public class **CBLEException**

extends Exception

Constructor Index

• **CBLEException(String)**

Basicexception.

Method Index

• **unimplemented()**

ThrowaRuntimeExceptionforunimplementedmethods.

Constructors

● **CBLEException**

Basicexception.

Methods

● unimplemented

```
public static void unimplemented()
```

ThrowaRuntimeExceptionforunimplementedmethods.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Class

com.adobe.cbl.CommitFailed_CBLError

java.lang.Object

|
+----java.lang.Throwable
|

|
+----com.adobe.cbl.CBLError

|
+----com.adobe.cbl.CommitFailed_CBLError

public class CommitFailed_CBLError

extends CBLError

Constructor Index

• CommitFailed_CBLError(String)

Constructsanexceptionforfailurestoopen.

Constructors

• CommitFailed_CBLError

Constructsanexceptionforfailurestoopen.

Class

com.adobe.cbl.OpenFailed_CBLErrorException

```
java.lang.Object
|
+----java.lang.Throwable
      |
      +----java.lang.Exception
            |
            +----com.adobe.cbl.CBLErrorException
                  |
                  +----com.adobe.cbl.OpenFailed_CBLErrorException
```

```
public class OpenFailed_CBLErrorException
```

```
extends CBLErrorException
```

Constructor Index

• **OpenFailed_CBLErrorException(String)**

Constructs an exception for failure to open.

Constructors

• **OpenFailed_CBLErrorException**

```
public OpenFailed_CBLErrorException(String msg)
```

Constructs an exception for failure to open.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Class

com.adobe.cbl.UnsupportedFeature_CBLEException

```
java.lang.Object
|
+----java.lang.Throwable
|
+----java.lang.Exception
|
+----com.adobe.cbl.CBLEException
|
+----com.adobe.cbl.UnsupportedFeature_CBLEException
```

```
public class UnsupportedFeature_CBLEException
```

```
extends CBLEException
```

Constructor Index

UnsupportedFeature_CBLEException(String)

Constructsanexceptionforunsupportedfeatures.

UnsupportedFeature_CBLEException

```
public UnsupportedFeature_CBLEException(String msg)
```

Constructsanexceptionforunsupportedfeatures.

Class

com.adobe.cbl.LimitCheckException

```
java.lang.Object
|
+----java.lang.Throwable
      |
      +----java.lang.Exception
            |
            +----java.lang.RuntimeException
                  |
                  +----com.adobe.cbl.LimitCheckException
```

public class **LimitCheckException**

extends RuntimeException

Field Index

- **limit**

- **msg**

Instancevariables

- **num**

- **OVER**

Upperlimitexceeded.

- **UNDER**

Lowerlimitexceeded.

• violation

Constructor Index

• `LimitCheckException(String,int,String,int)`

Constructsanexceptionforanumthatexceedsalimit.

Fields

• **OVER**

Upperlimitexceeded.

• **UNDER**

```
public static final String UNDER
```

Lowerlimitexceeded.

• **msg**

```
public final String msg
```

Instancevariables

• **violation**

```
public final String violation
```

• **limit**

```
public final int limit
```

• **num**


```
public final int num
```

Constructors

● LimitCheckException

```
public LimitCheckException(String msg,  
                           int limit,  
                           String violation,  
                           int num)
```

Constructs an exception for a num that exceeds a limit.

Parameters:

msg- An explanatory message.

num- Number that violates limit.

violation- Text describing violation. You are encouraged to use constants defined in this class, e.g., `LimitCheckException.OVER`.

limit- Max or min limit.

Interface com.adobe.cbl.TriDGfxCModel

TriDGfxCModel

extends `GfxCModel`

TriDGfxCModel defines 3D graphics content.

See Also:

`CModel`, `GfxCModel`

Field Index

• **LIGHT**

Identifies a light source.

• **POLYGON**

Identifies a polygon in 3-space.

• **TRIDMODEL**

Identifies 3D model.

• **VERTEX**

Identifies a vertex.

Fields

• **TRIDMODEL**

```
public static final String TRIDMODEL
```

Identifies 3D model.

● VERTEX

```
public static final String VERTEX
```

Identifies a vertex.

● POLYGON

```
public static final String POLYGON
```

Identifies a polygon in 3-space.

● LIGHT

```
public static final String LIGHT
```

Identifies a light source.

Interface com.adobe.cbl.TxtStylesLModel

TxtStylesLModel

extends [StylesLModel](#)

TxtStylesLModel defines text presentation information.

See Also:

[LModel](#)

Field Index

• CHAR

Identifies a character style.

• PARA

Identifies a paragraph style.

Fields

• CHAR

Identifies a character style.

• PARA

Identifies a paragraph style.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

```
java.lang.Object
|
+----com.adobe.cbl.VSiteEnumeration
```

public class VSiteEnumeration

extends Object

implements SiteEnumeration

implements a VectorbasedSiteEnumeration.

Field Index

- **sites**

The storage vector.

Constructor Index

- **(Vector)**

Construct with a Vector.

Method Index

- **getArray()**

Get the Site enumeration as an array.

Always returns the full list of Sites, even if `nextElement()` has already been called on this instance.

- **hasMoreElements()**

Moreelements?

- **nextElement()**

Getthenextelement.

- **nextSite()**

GetthenextSite.

- **resetEnumeration()**

Resetenumeration.

Oncetheenumerationisexhausted,thisobjectwillcontinuetothrowNoSuchElementException untilitisreset.

Fields

- **sites**

```
public final Vector sites
```

Thestoragevector.

Constructors

- **VSiteEnumeration**

```
public VSiteEnumeration(Vector sites)
```

ConstructwithaVector.

Parameters:

sites-TheinitialVectorofSites.AllelementsoftheVectorarerequiredtobeSiteobjects, andsitesmusthavepositivesize.

Methods

• hasMoreElements

```
public boolean hasMoreElements()
```

More elements?

• nextElement

```
public Object nextElement() throws NoSuchElementException
```

Get the next element.

Returns:

The next element.

Throws: NoSuchElementException

No more elements left.

• nextSite

```
public Site nextSite() throws NoSuchElementException
```

Get the next site.

Returns:

The next site.

Throws: NoSuchElementException

No more sites left.

• getArray

```
public Site[] getArray()
```

Get the site enumeration as an array.

Always returns the full list of sites, even if `nextElement()` has already been called on this instance.

Returns:

Array of Site. Returns null if there are no elements in the enumeration, or it can't be rendered as an array.

● resetEnumeration

```
public void resetEnumeration()
```

Reset enumeration.

Once the enumeration is exhausted, this object will continue to throw `NoSuchElementException` until it is reset.

[All Packages](#) [Class Hierarchy](#) [This Package](#) [Previous](#) [Next](#) [Index](#)

Interface

com.adobe.cbl.VectorGfxCModel

VectorGfxCModel

extends GfxCModel

VectorGfxCModel defines vector graphics content.

See Also:

CModel, GfxCModel

Field Index

- **CURVE**

Identifies a curve.

- **LINE**

Identifies a line.

- **PATH**

Identifies a path.

- **POINT**

Identifies a point.

Fields

- **PATH**

Identifies a path.

● LINE

Identifies a line.

● CURVE

```
public static final String CURVE
```

Identifies a curve.

● POINT

```
public static final String POINT
```

Identifies a point.

Interface

com.adobe.cbl.VideoDynCModel

VideoDynCModel

extends `DynCModel`

defines `videocontent`.

See Also:

`CModel`

Field Index

• TRACK

Identifies a video track.

Fields

• TRACK

`public static final String TRACK`

Identifies a video track.
